Project: Homework Tracker

Github link:
https://github.com/jaynair19111/homework_web

Proposal and Design

What the project is for:
I have created a website, designed and targeted to high schoolers like me. Its primary purpose is to keep track of study and progress easily online in a simple and clean way. It consists of a login page, a home page as an introduction to the website as a bit of information to get started and the main database for storing study tasks for different subjects, tasks and dates.

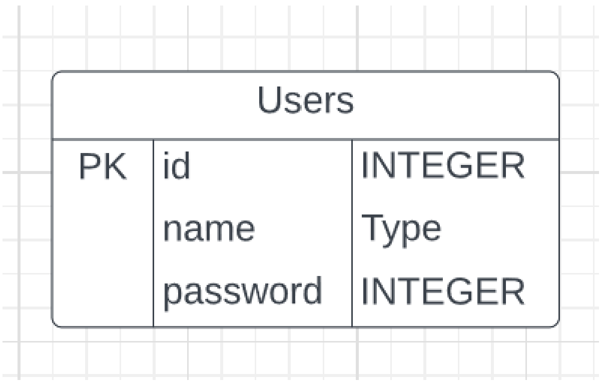
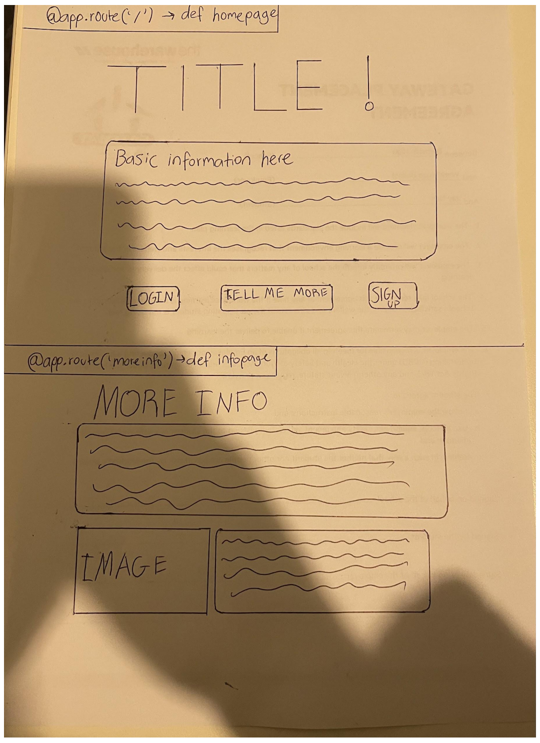
Requirements
- It must be able to be easy to use
- The database must be able to be modified by the user.
- It must be simple to avoid distracting the user.
- It must be clear when showing contained data and how to delete or add data etc.

Specifications

Resources
- I have access to multiple computers with tools I require.
- I have some skills and knowledge of required languages for the project such as python, html, css and SQL queries.
- I have had previous experience with required tools like visual studio, flask and SQLiteStudio.
- I have knowledge of different subjects in the school that should be included in the project.

Primary goals
- Let the user be able to add edit and delete homework tasks
- Have the user login or create an account if they do not have one

Starting Design Plan

 Users PK id INTEGER name Type password INTEGER	For my first design of the database I started off with a diagram of my database to get a good plan and hold of my idea for my project using lucid charts. This diagram shows the table and components required. This diagram would then go on to quickly help me create the actual database using SQLiteStudio.
 @app.route('/') → def homepage TITLE! Basic information here LOGIN TELL ME MORE SIGN UP @app.route('moreinfo') → def infopage MORE INFO IMAGE	I designed a simple layout of some of some possible routes that my website would have, and what I plan my website to look like. I drew this layout using a pen and paper as it would be much easier to put down my ideas on pen and paper rather than on a computer, and I could then later on use this layout plan to be created for my actual website for my project.

Implications

Functionality
The project must work correctly and as intended. A nonfunctional website means there is no point in using it in the first place and it cannot do anything. If something does not work as intended, then it can become frustrating to the user which is why everything that the user can do should be explained through button text (sign up) and the button should take you to the sign up page and not the home page.

How I plan to apply this to my project
- Testing the website to ensure that it works
- Always checking for errors
- All buttons and text are correct and work as intended.
- Have friends or family test the website to look for errors that I've missed.

Usability
Usability is how easy the project is to use by the user and how easy it is to navigate or do said task that the project is intended for.

How I plan to apply this to my project
- Making buttons clear on what they do
- Extra information and help if the user needs it
- Website layout is simple and easy to navigate

Aesthetic
Aesthetics are when the product looks good or feels good to use. The project should look nice and organized and straightforward to let the user know what's going on and allow them to easily access the website without getting frustrated or confused.

How I plan to apply this to my project
- Make the website very simple
- Avoid adding disorganized text or buttons
- Making sure everything on screen in the project has a purpose and no waste content is shown.
- Make buttons satisfying to click on
- Make everything neat and organized.

Health and Safety
The health and safety implication ensures that the project does not harm the users in any way or put them in any risk of possible danger. Things that could cause danger to the user are bright flashing colors which can put eppileptic users at risk.

How I plan to apply this to my project
- Avoid using very bright colors

- Avoid flashing colors or animations for buttons or backgrounds

SPRINT 1

Sprint 1 plan:

I plan on creating a login and sign up system for this sprint using a database that stores users, and the login being able to search through the database in order to login as a specific user while the sign up is made for creating an account in the database. This system will help me for the future sprints for adding personal tables for each account for personal tracking of homework. I also want to test the systems after I have donea

<u>Testing</u>

Query testing table

Purpose	Query	Pass/Fail
Search through the user database with entered credentials by the user that is the same as existing credentials in the user database.	`query = "SELECT name,password FROM users where name= '" + \ name+"' and password='"+password+"'"`	Pass
Adds a new user to the user database.	`sql1 = "INSERT INTO users(name,password) VALUES (?,?)"`	Pass

Programming testing tables

Login System Testing

What I am testing	Input	Expected results	Pass/Fail
Logging in with a user that exists in the user database.	Username: Jay Password: Jay	Takes me to the logged in page	Pass
Logging in with a user that does not exist in the user database	Username: iniwifnwinfowew Password: nwinfowinvwnepv	Refreshes the page as no user is found	Pass
Logging in with a user that does not exist and contains special characters	Username: @@#$%^&*()(*&^ Password: !~!@#$%^&*^*(*(	Refreshes the page as no such user exists without breaking	Pass
Logging in with no input	Username: Password:	Refreshes without breaking as no user exists	Pass
Entering an account and then going to another page	Username: Jay Password: Jay	Returning to the page removes the previously entered password	Pass

Sign Up System Testing

What I am testing	Input	Expected results	Pass/Fail
Creating an account with normal letters	Username: Jaynair Password: monkey123	Creates the user and notifies the user that the account is created.	Pass
Creating an account with special characters	Username: !@#$%^&*() Password: !@#%^&*(	Declines the users input and refreshes	Fail - website errors
Create an account that already exists in the system	Username: Jay Password: Jay	Declines users input and refreshes	Fail - creates another copy of the account causing database issues.
Entering nothing in the forms to create an account	Username: Password:	Declines users input and refreshes	Fail - Creates an empty account in the user data
Creating an account in all caps.	Username: MONKEY Password: MONKEY123	Creates an account as usual	Pass

Sprint 1 Summary

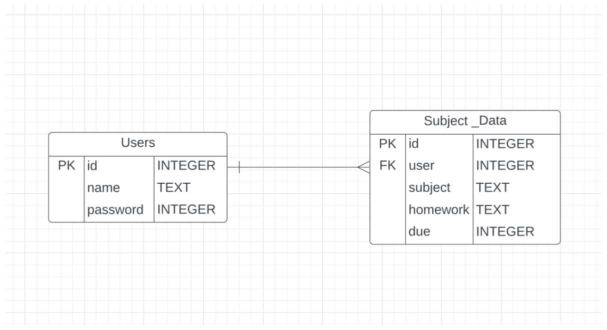
I have managed to create a simple login and signup system using a database to store accounts, and I did a mix of many basic but important functional testing of these systems. The login passed all of my testing but it is clear that the sign up system requires a lot more fixing as it is more prone to breaking and causing issues in the user database such as creating copy accounts and empty accounts.

SPRINT 2

Sprint 2 plan:

I plan on creating the tables connected to the user table in this sprint so that users can login and have their own personal table to edit and use to track their homework. In this sprint I will create that and do lots of functional testing to ensure that it works and is unlikely to break.

New database design

 Users PK id INTEGER name TEXT password INTEGER Subject _Data PK id INTEGER FK user INTEGER subject TEXT homework TEXT due INTEGER	This is my new database design for my second sprint which will connect the user id's with the subject data allowing to display which homework dates and subjects are assigned to.

Query testing table

Purpose	Query	Pass/Fail

Search through the user database with entered credentials by the user that is the same as existing credentials in the user database.	`query = "SELECT name,password FROM users where name= '" + \`
	`            name+"' and password='"+password+"'"`
		Pass

Adds a new user to the user database	`sql1 = "INSERT INTO users(name,password) VALUES (?,?)"`	Pass
Displays the subject_data table.	`sql = "SELECT * FROM subject_data"`	Pass
Adds a row with values to the columns in subject_data from 4 ordered inputs of the user.	`sql = "INSERT INTO subject_data(name,subject,homework,due) VALUES(?,?,?,?)"`	Pass
Deletes a row	`sql = "DELETE FROM subject_data WHERE id=?"`	Pass

Program testing table

Login System Testing (same as last sprint as no errors)

What I am testing	Input	Expected results	Pass/Fail
Logging in with a user that exists in the user database.	Username: Jay Password: Jay	Takes me to the logged in page	Pass
Logging in with a user that does not exist in the user database	Username: iniwifnwinfowew Password: nwinfowinvwnepv	Refreshes the page as no user is found	Pass
Logging in with a user that does not exist and contains special characters	Username: @@#$%^&*()(*&^ Password: !~!@#$%^&*^*(*(	Refreshes the page as no such user exists without breaking	Pass
Logging in with no input	Username: Password:	Refreshes without breaking as no user exists	Pass
Entering an account and then going to another page	Username: Jay Password: Jay	Returning to the page removes the previously entered password	Pass

Sign Up System Testing (fixed 2 errors from sprint 1)

What I am testing	Input	Expected results	Pass/Fail

Creating an account with normal letters	Username: Jaynair Password: monkey123	Creates the user and notifies the user that the account is created.	Pass
Creating an account with special characters	Username: !@#$%^&*() Password: !@#%^&*(	Declines the users input and refreshes	Fail - creates an account with special characters
Create an account that already exists in the system	Username: Jay Password: Jay	Declines users input and refreshes	Pass
Entering nothing in the forms to create an account	Username: Password:	Declines users input and refreshes	Pass
Creating an account in all caps.	Username: MONKEY Password: MONKEY123	Creates an account as usual	Pass

Sign up and login sprint 2 summary

The login system was fine and passed in sprint 1 making me leave it untouched for sprint 2. However for the sign up page I added a null and unique constraint allowing me to pass the sprint 1 errors but special characters are still an issue.

Main homework table testing

What I am testing	Input	Expected results	Pass/Fail
Adding a row	3, Game Design, Dev log, 21/11/22	Adds the row to the table	Pass
Deleting a row	Clicking the delete button next to a row	Deletes the selected row	Pass

Testing summary

Overall for my testing I was able to easily figure out problems and what I needed to work on with my testing tables throughout the sprints. Such as in sprint 1 I tested new inputs such as copies of the same account which would be added which can cause errors, and this made me know that I had to fix the issue for the next sprint and how I learned and came across the unique constraint. I tested many possible ways to break the website and would see if the intended outcome would appear, and if not I would try to learn and fix the issue and test it again until I have resolved the problem. Overall I think I did well testing but I believe some areas such as the sign up page could've been done with more testing to resolve some issues I wasn't able to finish for my final outcome.

Relevant implications summary

Did I apply the relevant implications I mentioned at the start of my project?

Functionality

- Testing the website to ensure that it works - Through many testing tables through different sprints I was able to locate errors of each sprint and what I needed to fix and work on for the next sprint and this allowed me to make sure I have ensured that the website works.

- Always checking for errors - I would test out the website for any possibility that can break the site so that I can fix any errors that occur or bugs that are found.

- All buttons and text are correct and work as intended. - Lots of testing of buttons made me sure that they work and I also made sure that there were no spelling errors or wrong wording that could make the buttons misleading.

- Have friends or family test the website to look for errors that I've missed. - Friends can find errors better than I can and they are more likely to find errors and bugs in my website. For example I tested my sign up page with Ryan and that is the reason I found out about null and unique constraints to avoid random unintentional accounts to be made that can break the website.

Usability

- Making buttons clear on what they do - Similar to functionality I made sure no spelling mistakes were made or incorrect words to not make the buttons misleading which can cause anger and usability issues.

- Extra information and help if the user needs it - The home page contains basic information on how to get the user started.

- Website layout is simple and easy to navigate - The buttons are always clear and right in the middle to let the user know what to do easily.

Aesthetic

- Make the website very simple - I followed this idea and made sure my website was very simple and easy to navigate.

- Avoid adding disorganized text or buttons - I made sure buttons were laid out how they should be without being all over the place.

- Making sure everything on screen in the project has a purpose and no waste content is shown. - I removed many things on the website that were not needed.

- Make buttons satisfying to click on - I added hover effects on buttons to make them more aesthetically pleasing.

- Make everything neat and organized. - My website was laid out neatly and symmetrical to be pleasing.

Health and safety

- Avoid using very bright colors - For my background which had lots of color, I tried to avoid making it very bright and dimmed down the shades of the colors.

- Avoid flashing colors or animations for buttons or backgrounds - I made sure that the gradient animation on the background was slow and smooth instead of flashing which can be dangerous and not buttons had any quick or flashing animations.

Overall summary and thoughts on my final outcome

Unfortunately I wasn't able to make my website as advanced as I wanted to be, but what I have managed to build was a website with a simple login and sign up system as well as a working database that you can add, delete and edit homework tasks, dates and subjects. If I was to do this again, I would start my project at a much much earlier date and be more organized, and I would've made the login system more functional by allowing the logged in user to automatically have a session and their logged in user id automatically placed in what homework task they added automatically rather than manually putting in their username when they have already logged in, making the login system feel underwhelming for this project. But something that I thought I did well was styling of the website and making navigation of the website clear and simple as well as minimizing the chances of errors or the site breaking as much as possible.