MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG445 Software Development with Scripting Languages

Assignment 3: A Game Platform Website

This assignment aims to help you practice databases, HTML, CSS, browser-side scripting with JavaScript, and server-side scripting with Python Flask[1]. Your main task in this assignment is to develop a website for people who want to publish the video games they developed.

You are expected to use HTML to define the content of web pages, CSS to design the web pages, JavaScript for browser-side scripting for **web form validation purposes where appropriate**, and Python Flask for server-side scripting. You are also expected to use an **SQLite** database to store and manage data.

Overview

In this assignment, you are expected to develop a video game platform. People first need to register to publish their games. These published games will be publicly visible which means that there is no need to log in to see and search for the published games. Some users will also be admins who can manage some aspects of the website.

Functionalities

The website should have the following functionalities for people who want to use the platform:

- **Registration:**

  The home page should have a link to access another page that includes a web form for people to register. They should be able to register with the following details: (1) username, (2) password, (3) full name, and (4) email address. All these details are mandatory.

  The username should be unique for each person, and the password should include at least one upper case letter, one lower case letter, and one digit and its length should be at least ten. Additionally, if the email address ends with "@game.metu.edu.tr", the user will be created as an admin.

  If there is any problem, they will be directed to the registration page again and the problem will be shown to the user just the under the form. Otherwise, they will be directed to a page which shows the confirmation message and includes a link to go back to the home page.

- **Login:**

  The **home page** should have a web form to allow people to log in with their usernames and passwords. If there is any problem (such as invalid username, incorrect password, etc.), the problem will be shown to the user just under the form. Otherwise, the login form will be removed from the home page, and the following menu will be displayed for users:

  Home Page | Published Games | My Profile | Logout

  or the following menu for admins:

  Home Page | Manage Genres | My Profile | Logout

[1] https://flask.palletsprojects.com/en/stable/

- **Logout:**

  After login, the home page should also have a link for people to log out from the website (see above). Once they log out, they will be directed to the home page again.

- **Published Games:**

  After login, the home page should have a link for users to manage their games. This page should include a web form where they can create a new game with the following details: (1) title, (2) price, and (3) genres (Since there can be multiple, you can present all genres as check boxes). These details are mandatory. The users can also add extra details that are not mandatory: (4) description, and (5) whether the game is full release (as opposed to early access games). You need to use radio buttons to allow the user to select whether the game is full release.

  A unique identifier will be automatically assigned to each game using AUTOINCREMENT in SQLite. If there is any problem, the problem will be shown just the under the form. If a game belongs to multiple genres, then all of these genres will be shown separated by a comma (such as Genre1, Genre2).

  This page will also list the previously published game of the person who has logged in. The user should also be able to delete his games when s/he clicks on the corresponding Delete button.

Title	**Price**	**Genres**	**Delete**
…	…	…	Delete
…	…	…	Delete

- **Manage Genres:**

  After login, the home page should have a link for admins to manage genres. This page should include a web form where they can create new genres.

  A unique identifier will be automatically assigned to each genre using AUTOINCREMENT in SQLite. If there is any problem, the problem will be shown just the under the form.

  This page will also list all current genres on the platform.

Genre Name
…
…

- **My Profile:**

  This page displays the details of the profile of the person who has logged in in a web form and allows the user to change the password by clicking the **Change Password** button.

Username	…
Password	••••••••••••••
Name	…
Email	…

Change Password

The website should also allow **all people** to search for published games as explained below.

- **Search:**

  The **home page** should include a text box with a button which allows searching for published games based on keywords. It will also include a combo box with all the available genres to select the genres. There should be also an option to consider all genres.

  The games which have any field that includes at least one of the keywords will be listed. Please note that it does not have to be a full match, so if the keyword is "banish", and the title is "road of banishment", then it should be listed as well.

  If a particular genre is selected, the games are listed in a table, or "No games" will be displayed if no game is found. If all genres are selected, then the matching games should be categorised based on the genres as shown below, and "No games" should be displayed for a genre which has no matching games. If a game belongs to multiple genres, then they should be listed under each of those genres. Note that the genres presented are just an example.

  Role-Playing:

Title	**Description**	**Genre**	
…	…	…	See More
…	…	…	See More

  Real-Time Strategy:

Title	**Description**	**Genre**	
…	…	…	See More
…	…	…	See More

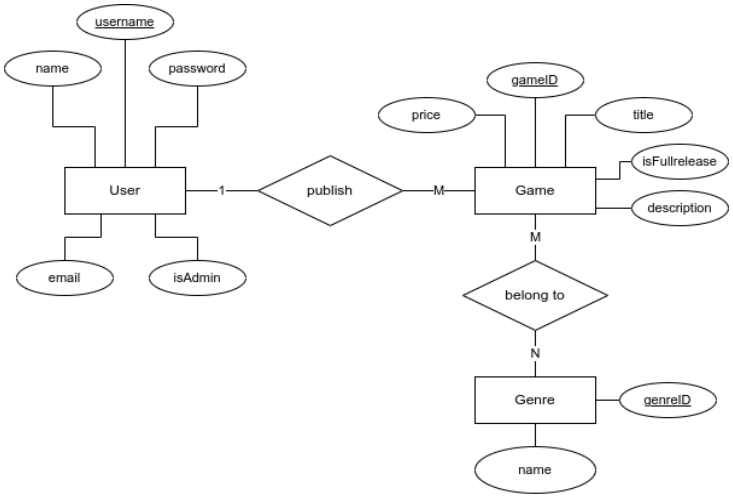
- **See Selected Games:**

  When people click a link called "See More", they will be directed to a page which shows all the details of the selected game as shown below. This page should also include a link to go back to the home page.

Title	…
Description	…
Genres	…
Full Release	[Yes | No]
Price	…

Database

You need to create an **SQLite** database for this website based on the following entity-relationship diagram (ERD). You are expected to provide a separate Python script called **dbscript.py** to create the necessary tables.



Rules

- You need to write your program by using **Python 3.x**.
- You can **only** use all built-in functions and modules, apart from Flask.
- You need to create your CSS and JavaScript files as external files and include your HTML pages.
- **You are not allowed to use any design template.**
- You also need to create a file called **ReadMe.txt** which contains the following items. Please note that **if you do not submit ReadMe.txt, your submission will not be evaluated.**
    - Team members
    - Which version of Python 3.x you have used
    - Which operating system you have used
    - How you have worked as a team, especially how you have divided the tasks among the team members (who was responsible for what?), how you have communicated, how you have tested the program, etc.
- You need to put all your files (including your database file with the .db extension) into a folder that is named with your student id(s) and submit the compressed version of the folder in the **.zip** format.
- **Only one team member** should submit the assignment.
- **Code quality, modularity, efficiency, maintainability, and appropriate comments** will be part of the grading.

Grading Policy

The assignment will be graded as follows:

Grading Item	Mark (out of 100)
Database Creation Script	5
Login, Logout and Session Management (Such as, not allow people to access unauthorised pages)	10
Register	5
Add game	5
Access games	5
Delete game	5
Manage genres	10
Manage Profile	10
Search games and list them categorised by category	20
See the details of the selected game	10
Navigation within the website	5
Form validation with JavaScript	5
Design with CSS	5