COMPSCI 235 S2 2023

Assignment Phase 3

15% of your grade

Due date: 11:59 pm Thursday 21tst September

Total – 35 marks

Objective

The purpose of Assignment phase 3 is to add new features to the Flask web app you built in Assignment phase 2. The focus will be on enhancing user experience through the addition of features like user authentication, game reviews, user profile and a wishlist feature. You'll also improve the app's quality by adding more unit tests and end-to-end (or integration) tests. The goal is to make the app more interactive and reliable.

Each team member is required to make their own commits to the shared GitHub repository to demonstrate their individual contributions to the project. A lack of commits will be interpreted as a lack of participation and effort.

IMPORTANT: For your code to be eligible for marks on the tasks specified in the functional requirements, it must adhere to the following non-functional requirements.

Non-Functional Requirements (same as Assignment phase 2)

1. Project Structure and Conformance **[1 mark]**
   - Conformance to established project structure as discussed in the lectures.
   - Ensure vital documentation, like '*requirements.txt*' and '*README.md*', is present.
   - The application should run by typing '*flask run*', from within the virtual environment in a terminal window.
   - The tests should run by typing '*python –m pytest –v tests*', from within the virtual environment in a terminal window.
   - If the application doesn't run properly, the remaining steps will NOT be marked.

2. Use of Blueprints **[1 mark]**
   - Register blueprints to segregate the application's components. For example: home, games, genres, search etc.
   - Each blueprint should encapsulate a distinct set of functionalities adhering to the principle of Single Responsibility.

3. User Interface & Design **[1 mark]**
   - Implement CSS for a visually captivating application.
   - Utilize Jinja templates to ensure a consistent appearance across pages while minimizing repetitive code.

4. Flask Routes and HTTP Protocol **[2 marks]**
   - Clearly define routes/URLs for different functionalities (e.g., /games, /genres, or /search etc.)
   - Ensure that the application uses the appropriate HTTP methods, query parameters, response codes etc.

5. Abstract Repository Interface **[2 marks]**
   - This interface should define the necessary methods for fetching, storing, and manipulating data about games.
   - For this assignment, you'll be using a memory repository as the data storage mechanism.

Functional Requirements (NEW!!!)

Task 1: User Authentication **[5 marks]**

- Implement a registration and login/logout system using Flask-WTF and password validator libraries (you'll need to add these to requirements.txt).
- Provide clear feedback to users when they have successfully logged in or logged out.
- Ensure the use of signed cookies and CSRF tokens for enhanced security.
- Users, once logged in should be able to provide reviews/ratings for games. You'll implement the @login_required decorator to restrict access to such functionalities. The @login_required decorator should be placed before every function that requires user authentication (See login_required() function in covid/authentication/authentication.py).

Task 2: User Reviews – Comments and Ratings **[6 marks]**

- Allow authenticated users to post a review and give a rating for each game.
- Reviews and average rating should be displayed on each game's detail page. Display these reviews in a user-friendly manner, ideally sorted by most recent or other relevant criteria.

Task 3: Wishlist or Favourites Feature **[6 marks]**

- Allow authenticated users to add games to their Wishlist or Favourites list.
- The wishlist/favourites should be easily accessible either through the user's profile page or a separate dedicated section within the web application.
- Allow users to remove games from their Wishlist or Favourites list.

Task 4: User Profile Page **[5 marks]**

- Design and implement a user profile page that is accessible only to authenticated users. This profile page should display an organized history of the user's activities, such as games they've rated, reviewed, or added to their wishlist.
- This task mostly involves creating a new view that collates and displays existing data relevant to each user.
- Implement CSS styling to make the user profile page visually appealing. The design should be consistent with the rest of the application but can have unique elements that make the profile page special.

Task 5: Unit Tests for New Features **[3 marks]**

- Extend the unit tests to cover all new functionalities introduced.
- It is important that the tests are written well and provide good coverage. These should be readable with meaningful names, and include comments detailing the test cases.

Task 6: End-to-End Testing **[3 marks]**

- Implement end-to-end tests to simulate complete user requests, such as registering, logging in, browsing games, adding games to the wishlist, logging out etc.
- It is important that the tests are written well and provide good coverage. These should be readable with meaningful names, and include comments detailing the test cases.

Submission:

- For marking, we will use the latest commit that you have pushed to the GitHub repository before the submission deadline. Please make sure your submission is pushed to the remote GitHub repository, and not just committed to your local version of the repository!
- Each webapp should display the name of the developers (the team members).
- When assessing your work, markers will expect to:
  - Find a README file in the project's root directory that explains how to set up a virtual environment and install any dependencies via a requirements.txt file
  - Run the application by typing '*flask run*', from within the virtual environment in a terminal window.
  - Run all tests by typing '*python –m pytest –v tests*', from within the virtual environment in a terminal window.
  - **Failure to meet these requirements will result in a FAIL grade.**

⚠ Sharing assignment solutions and source code does not help learning. Consequently, our academic integrity policy does not permit sharing of solutions or source code leading to solutions, nor does it allow sourcing solutions or source code from any third party. Violation of this will result in your assignment submission attracting no marks, and you may face further disciplinary action. Therefore, please do not share assignments, assignment solutions and/or source code leading to assignment solutions, or use material from others in your assignments. You must not publish assignments or solutions in any form online at any time. You will be liable if someone copies your solution. There are also copyright and IP issues. Please come talk to us if you have any doubt over what is legitimate and what is not.

You can refer to online tutorials, generative-AI tools (such as ChatGPT) and online resources. However, please learn from them and implement the solutions yourself based on what you learnt from these sources. You must not copy from these sources.

Don't leave your computers, devices, and belongings unattended — you must secure these at all times to prevent anyone having access to your assignments or solutions. If others are found to have used your solution, you will also face disciplinary action.