Homework #3 (10 points)

Purpose

The purpose of this assignment is to help you demonstrate your understanding of threat modeling, and Django security.

How to Submit it

Upload your deliverables on Gradescope.
You can zip all the files and upload the zip file (Gradescope will automatically extract the files upon submission).
If there is any large file in the submission, gradescope throws an error message. Please remove any large file prior to submission.
If the error persists, please contact the instructor immediately (attach your submission to the e-mail or share its Google Drive link).

Deliverables

- A PDF file with your answers to the conceptual questions in **Part 2 & 3** (it can be a combined PDF or a separate PDF for each part). **PDF file only** (do not upload in other format).
- The source code for the website with your implementation in **Part 1**
    - If you're using pipenv/virtualenv, make sure to ***not*** include the hidden folders with all the binaries for Python libraries (that would make the zip file too large for Gradescope to accept).

When submitting to gradescope, create a ***ZIP*** file and then upload this file to it.
DO NOT use another compression format (ex: rar, tar, etc)

Task

In this individual homework, you will identify threats, create a DFD, and implement ***fixes*** for the web application from **HW2**.
Recall that the application was a simple Web application where a user can add/delete/lists tasks (i.e., a task management app). Refer to HW2 for instructions on how to run the application.

Part 1: Fixing Issues from HW2

In HW2, you played the role of a *software consultant* hired to identify the security vulnerabilities in a Web application. In this homework, you will now play the role of the *software developer* that will fix the identified security vulnerabilities.

- **Q1:** Change the code provided to fix ~~all~~ ***at least two*** of the vulnerabilities you found in the web application.
- **Q2:** Include in the code comments with an explanation about the fix.

Part 2: Identification of Threats

Trust Levels		
ID	**Name**	**Description**
1	Anonymous Web User	A user who has connected to the website but has not provided valid credentials.
2	User with Valid Login Credentials	A user who has connected to the website and has logged in using valid login credentials.
3	User with Invalid Login Credentials	A user who has connected to the website and is attempting to log in using invalid login credentials.
4	Website Administrator	The Website administrator can configure the website.

Assets			
ID	Name	Description	Trust Levels
1.1	User Login Details	The login credentials that a user has to log into the website.	(2) User with Valid Login Credentials (4) Website Administrator
1.2	Task Details	The task details stored by the database.	(1) Anonymous Web User (2) User with Valid Login Credentials (3) User with Invalid Login Credentia (4) Website Administrator

Entrypoints			
ID	**Name**	**Description**	**Trust Levels**

1.1	Index Page	The index page for the website is the entry point for all users.	(1) Anonymous Web User (2) User with Valid Login Credentials (3) User with Invalid Login Credentia (4) Website Administrator
1.2	Add task page	The page where users can add tasks.	(2) User with Valid Login Credentials (4) Website Administrator
1.3	Login page	The login function accepts user supplied credentials and compares them with those in the database.	(2) User with Valid Login Credentials (3) User with Invalid Login Credentia (4) Website Administrator
1.4	Delete task page	The page used to delete a task.	(2) User with Valid Login Credentials (4) Website Administrator

Q1) Consider that we have the (simplified) list of entry points, trust levels, and assets above for the application provided in the HW2 as shown in the tables above. Write down 1 threat for this application using the threat model profile shown in class:

ID	*0001*
Name	*Unauthorized Task Deletion via Direct URL Access*
Description	*Deletion of tasks occurs without any sort of checks - including authentication and ownership of task*
STRIDE Classification	*Elevation of Priviledge - users can delete tasks belonging to other users, which shouldn't be possible as a standard user*
Mitigated?	*Yes - now checks if user is authenticated, if task belongs to user, and HTTP method is POST only*
Known Mitigations	*Authentication check, confirming user ID == task user ID, POST only*
Entry Points	*1.4 Delete task page (/tasktracker/delete/{taskID})*
Assets	*1.2 Task Details*

Q2) Create a data flow diagram for ***one feature of your choice*** of the HW2's application.

Part 3: Research Synthesis

Graduate students are expected to have evidence of *research synthesis* during the semester. To fulfill this requirement, please read the following paper and answer the questions below:
Mai, Phu X., et al. "Modeling security and privacy requirements: a use case-driven approach." Information and Software Technology 100 (2018): 165-182.

1) **Summarize the lessons learned while the authors applied their approach to an industrial healthcare project and from the interviews with engineers.**
   Answer will be roughly 1 page in length.

Grading Rubric

The HW will be evaluated based on the following criteria:

Part 1 4.5 pts / CSE60770	Is the implemented solution correctly fixing the identified vulnerabilities? Is the code free from runtime errors?
Part 2 4.5 pts / CSE60770	Are the question answers showing applicable security threats? Is the diagram using the DFD notation shown in class? Is the DFD depicting correct data flows?
Part 3 1 pt / CSE60770	Are the questions answered with enough detail?

Username: `user1` / **Password**: `p*2sfHQP==`
Username: `user2` / **Password**: `V?vZT+MD4e`

In the paper's application of the RMCM approach, they surveyed 4 engineers within a large healthcare application, with a wide variety of sources across multiple platforms and user roles. The healthcare application was designed to gamify user health for the elderly, including social interactions and exercise. The authors intensively interviewed engineers within the project, with all of them having familiarity with use case driven development and modeling, as well as at least a couple of years of experience in industry.

From the survey results, we can see a few key points that this paper has touched on. Results are scored from 1-4, with 1 being disagree and 4 being agree. The first thing I notice are the scores for questions asking about communication between engineers and shareholders (Questions 1, 4, and 12). These have scores of 2.5, 2.75, and 2.25 respectively. It appears that the developers question the ability for this process to accurately display security vulnerabilities and use cases at a level that will satisfy both technical developers and non-technical stakeholders. The only other score of 2.25 is Question 7 (Mitigation schemes are simple enough to enable communication between analysts and programmers), which is quite similar to the

engineer / stakeholder dynamic. Scores > 2 are still above neutral and are positive, but these are some of the lowest scores for all of the questions. This was only a survey of 4 people, and the stereotypical engineer disdain for non-technical managers may be present, but this shows some level of miscommunication or misunderstanding between the engineers and stakeholders that still may be present despite this project's attempt to minimize it.

Some of the higher scores arise from questions regarding the method's ease of use and ability to accurately capture threats and report them to the developer. Questions 2 (If a misuse case diagram like the one we presented were available to you, would you use it to help you capture or understand security threats and mitigations?), 3 (The notation provides enough expressiveness to conveniently capture the security threats and mitigations in your projects), 6 (Security threats captured in the misuse case diagram are adequately reflected in the specifications), and 9 (The steps in our modeling method are easy to follow) were the highest scores recorded, ranging from 3 to 3.5. This shows that the method proposed does very well at visualizing the security threats present and relaying them to the user in an intuitive way that they would use again.

Looking at the practical application results from EDLAH2, the numbers tell a compelling story about what the method actually accomplished. They modeled 17 misuse cases that threatened 9 use cases, with 20 total threaten relationships between them. This shows that each use case had to deal with multiple security threats, reinforcing why a systematic approach is necessary. This was particularly relevant for this application, as they have 10 developers in 3 teams, with each team being responsible for their own components that may threaten other teams' components.

At the end of the interviews, the selected developers had a few comments about the method's feasibility and implementation. The biggest concern was about widespread adoption. Even though developers liked the method and found it was relatively easy to understand and learn with instruction, they expressed skepticism about introducing yet another modeling tool that would require both developer approval and stakeholder funding. They did find the automation and consistency of the tool to be very useful. They also pointed out that dynamic updates were required to keep up with modern technologies and vulnerabilities. Overall, the developers seemed to be cautiously optimistic about the methodology.