App Name: Spartan Sync
Andrew Dover, Aryan Gaur, Andrew Porasl
Repo: https://github.com/adover06/CMPE131-LMS

Problem Statement:

Current LMS tools like Canvas can be very overwhelming and confusing for students and instructors to use, leading to things such as missed assignments, announcements, and a bad user experience. A system that is simple to use, prioritizes tasks the same as a student would, and creates assignments intuitively for instructors is desperately needed.

Proposed Solution:

With Spartan Sync, there's an easy-to-use, more effective way of navigating coursework for the student, and an effective way of managing it for instructors. Instead of various different pages with cluttered home screens and ambiguous pages, SpartanSync is a lightweight, simple application that guides you to everything you need intuitively. Grades are displayed clearly, and announcements are separately shown and prioritized, all while retaining all necessary information. Removing the clutter minimizes the navigation friction.

In Milestones 2 and 3, we'll implement Flask-Login, SQLite, and WTForms. This will mean we'll have a functional login system, student dashboard, course page, assignment page, along with an assignment creation page for the instructors to publish assignments. Essentially, everything that's a UI prototype right now will be feature complete.

Key Users:

Students
- Needs to be able to access assignments, due dates, and grades
- Consistent and simple navigation across all courses

Instructors
- Clear organization of student submissions
- One-page, fast, easy assignment creation page

TAs
- Access to grading and reviewing submissions
- Limited access to modifying anything else regarding student grades and course materials

MVP
1. Add a login form with WTForms to control user access.
2. Main Student Dashboard: Assignments, classes, and a small nav menu.
3. Main Instructor Dashboard: Classes and nav menu

4. The Assignment List in the dashboard for students will show the assignment title, class, and point value.
5. Assignment Creation Form for the instructors to create assignments.
6. The main section of the home page will have an array of card elements to display classes.
7. The nav menu will be present on all pages except the login (HTML wrapper using Jinj2).
8. Roles will be split between Teacher and Student.

Stretch Goals

1. Each instructor can define an estimated time taken for an assignment. Then, on a "Student-type" questionnaire on registration, a student can indicate if they are more likely to prefer starting on the work that takes the longest, or the shortest first. This way, the assignment to-do list will be prioritized by due date first, and preferred order by the duration of the assignment next.
2. A simple chat or email-style communication between students and instructors.
3. An AI time manager that summarizes and outputs a study plan based on the assignments due and the student questionnaire answered previously.

Success Criteria

1. All routes described, load and match the prototype designs without server errors. Each page will look like the UI screenshots given in M1.
2. Users can navigate through a complete workflow. This means students can log in, go to the dashboard, go to the course, assignment list, and assignment details. This means that instructors can log in, go to the dashboard, go to the course, and create an assignment form.
3. Navigation structure is consistent, complete, and intuitive. All the links work correctly and lead to the intended pages. No extra clutter is shown, and all course pages are extremely similar in format and use, with little to no navigational inconsistencies.

Contributions (current and present)

- **Andrew Dover**
    - Developed initial layout
    - Created login page
    - Created base for assignment turn-in and assignment creation (stubs)
    - Added a functioning login manager with Flask-Login session manager
    - Created Models for db login, and connected with the login + homepage
    - Worked on Tailwind styling for pages
    - Nav bar parent class HTML (Jinja2)
    - Implemented password hashing for the backend and database integrations.
    - Work on
- **Andrew Porasl**
    - Added Student, TA, and Instructor roles into the authentication system
    - Implemented the Course, Assignment, Rubric, Submission, and Announcement data models

    - Created templates for assignment lists, assignment creation forms, assignment details, announcements, announcement creation, dashboard, course list, course detail, and class management pages
    - Implemented status badges for assignments
    - Course cards, assignment overview, and announcements
    - Added routing logic connecting new templates to backend models
    - Cleaned repo structure by adding functionality to refresh the database every time the repo is initialized to make sure we have a proper environment.
    - Updated README with screenshots, role flows, and feature summaries.
- **Aryan Gaur**
    - Managed circular import issue and refactored code to maintain usability
    - Implemented correct use of blueprints
    - Implemented feature and login page
    - Created read me with screenshots and summary of program
        - Updated formatting and page setup
    - Teacher assignment creation and announcement feature
    - Observed all processes, designed tests for each process and created and implemented a comprehensive testing plan using pytest