ZotPlanner

**High-Level Project Description:** Our project is a web app designed to streamline yearly course scheduling for undergraduate computer science students at UCI. The app aims to consolidate essential resources and information in one place, reducing the need to access multiple platforms. By providing a centralized, intuitive interface, our tool will simplify the decision-making process for students planning their academic schedules.

**Target User Group and Context:** The primary users of this app are undergraduate computer science students at UCI, navigating the complex process of course planning each academic year. The context involves both academic and extracurricular considerations students face when choosing courses, such as balancing workloads, fulfilling prerequisites, and ensuring they’re on track for graduation. Our tool is designed to serve as a one-stop solution, integrating the information and features students typically seek.

**HCI Principles and Design Considerations:**

1. **User-Centered Design**: ZotPlanner puts the student’s needs and context at the forefron such as tracking remaining requirements, assessing workload, and selecting compatible professors.
2. **Perceivable Interface**: The layout will be a clean and emphasizes hierarchy, with DegreeWorks data displayed prominently, followed by personalized course recommendations. Visual cues (e.g., color-coded course statuses, icons for prerequisites) improve scannability. A contrast-friendly color palette and screen-reader compatibility will ensure that all elements are accessible.
3. **Operable Navigation**: ZotPlanner uses a simple, guided workflow to help students explore options while reducing decision fatigue. Key features like “My Remaining Courses” and “Suggested Schedules” are accessible with a single click. An optimized mobile view allows users to plan on the go.
4. **Understandable Information Architecture**: The interface is divided into clearly labeled sections like "My Requirements," "Available Courses," and "Professor Reviews." Each feature uses familiar academic terms, and interactive help text will offer explanations and guidance on tool usage.
5. **Robustness**: ZotPlanner integrates directly with DegreeWorks and Rate My Professor, updating data dynamically so that users have accurate, real-time information on course availability, prerequisites, and professor ratings. Robust error-handling will guide users if data fails to load, suggesting potential workarounds.

**Key Features:**

* **Integrated Degree Requirements**: See DegreeWorks data directly in ZotPlanner to track required courses and completed credits.
* **Smart Suggestions**: ZotPlanner suggests optimal course combinations based on user preferences, such as professor rating and weekly workload.
* **Rate My Professor Integration**: Professor reviews provide students with insight into teaching styles, course difficulty, and past student experiences.
* **Prerequisite Tracker**: Courses without prerequisites or required courses are visually marked, preventing registration conflicts and saving students time.
* **User Experience Optimization**: ZotPlanner's design includes common web accessibility standards to ensure all students, including those with disabilities, can use the app.

**Team Members:**

* Delphine Tai-Beauchamp
* [Agaton Julian Pourshahidi](mailto:ajpoursh@uci.edu)

**Templates and References:**

* Very similar to Zot4Plan but including prerequisite restrictions as well as course offerings (which will only apply to the current academic year)

**Additional Information and Feedback Request:** We would appreciate feedback on:

* The feasibility of integrating live data from Rate My Professor and DegreeWorks into a web app, as well as potential data privacy or access concerns.
* Whether there are additional HCI considerations or features we could add to improve accessibility and usability for our target users.

**Pseudocode Sample:**



// Inputs: User’s remaining courses from DegreeWorks, current prerequisites, course load preference

Function generateSuggestedSchedule(userProfile):

userRequirements = fetchUserDegreeRequirements(userProfile)

availableCourses = fetchCourseOfferings()

filteredCourses = filterAvailableCourses(availableCourses, userRequirements)

sortedCourses = sortCoursesByPriority(filteredCourses)

recommendedSchedule = scheduleBuilder(sortedCourses, userProfile)

return recommendedSchedule