Design Document

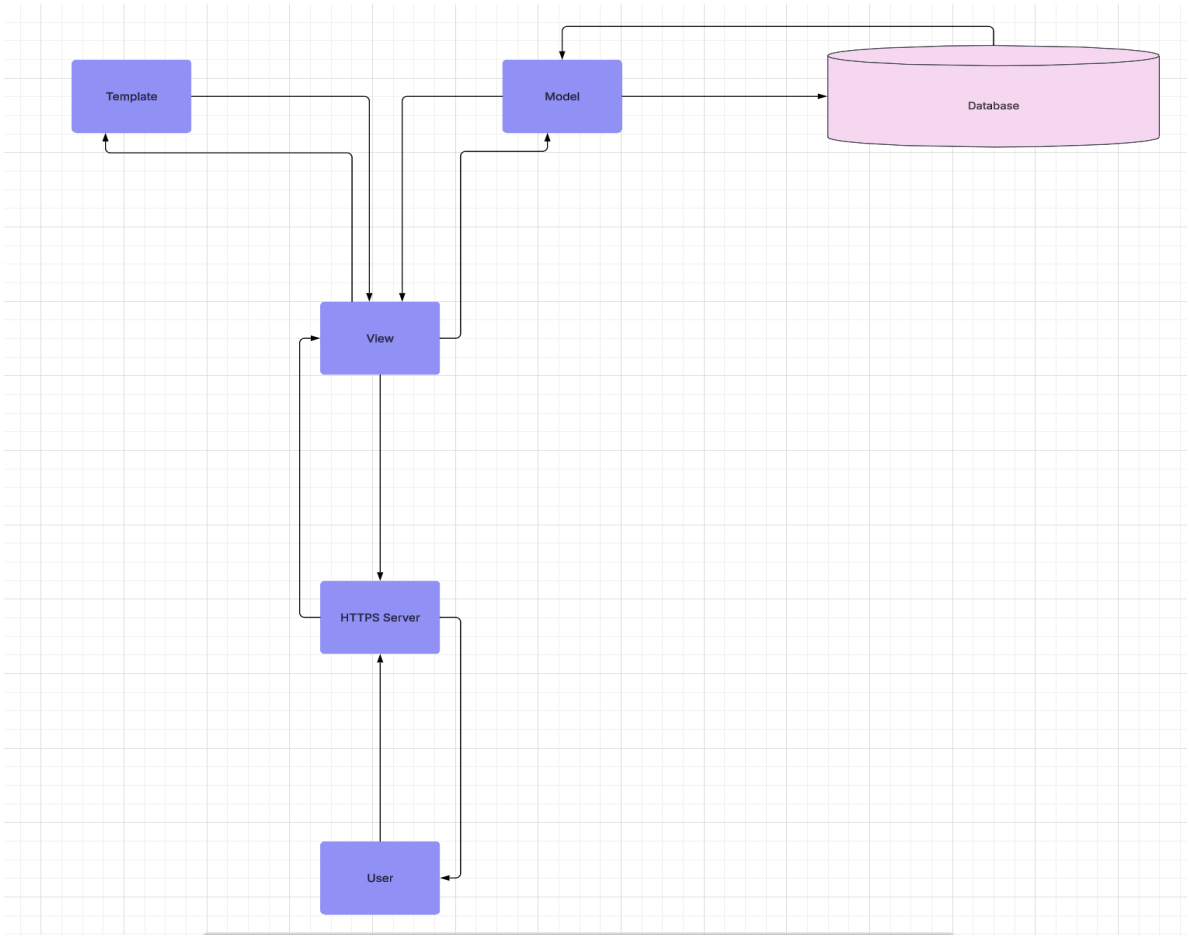
System Description: CourseWise™ will address the challenge of course demand prediction in academic settings. Currently, faculty and staff lack tools to accurately predict course enrollment needs, leading to either overcrowded courses or underutilized resources. The system will solve this by creating a web-based platform that analyzes historical Workday enrollment data to predict future course demand and present these insights through intuitive visualizations and reports.

Architectural Overview:

The system will follow a three-tier architecture:

1. Frontend: Web interface for data visualization and user interaction
2. Backend: API server handling business logic and predictions
3. Database: Storage for historical enrollment data and prediction results

System Diagram:



Functional Requirements:

- **Epic 1**: Predicting Course Enrollment and Class Availability
    - **Goal**: As a user of CourseWise, I want to know my likelihood of getting into a class based on my major, year, and course preferences.
- **Epic 2**: Understanding Class Demand and Difficulty
    - **Goal**: As a user of CourseWise, I want to access class demand and difficulty to make informed decisions about which courses to enroll in.
- **Epic 3**: Personalized Class Suggestions
    - **Goal**: As a user of CourseWise, I want to receive personalized class suggestions based on my student information and preferences to help find suitable alternatives.
- **Epic 4**: Interactive Course Exploration
    - **Goal**: As a user of CourseWise, I want to interactively explore course options and instantly receive predictions and suggestions for each class I click on.
- **Epic 5**: Class Scheduling with Degree Requirements
    - **Goal**: As a user of CourseWise, I want to see which courses are required for my degree and when they are offered, so I can prioritize courses that will help me graduate on time.

Non-Functional Requirements:

- Non Functional 1: Performance
    - Support for multiple concurrent users
    - Page load time under 3 seconds for visualizations
    - Smooth interaction with visualizations (no lag when filtering or updating views)
- Non Functional 2: Usability
    - Interface should be intuitive and require minimal training
    - System should be accessible on both desktop and mobile browsers
    - Clear visualization of data with appropriate labels and legends
    - Support for major browsers (Chrome, Firefox, Safari, Edge)
- Non Functional 3: Maintainability
    - Well-documented code with clear comments
    - Modular design for easy updates if needed
    - Comprehensive API documentation
    - Version control with clear commit history

Technologies and Frameworks:

- Programming Language: Python
- Database: Django/SQLite

Minimum Viable Product:

- Basic user interface showing historical enrollment data
- Simple prediction model for course demand
- Basic visualization of trends

Create a Preliminary Roadmap for the Semester:

- Sprint 0: Project Setup & Planning
    - Begin Workday analysis and planning
    - Create User Stories and list of requirements
    - Connecting with the Registrar office
- Sprint 1: Data Foundation
    - Upload data to a database
    - Create models for data
- Sprint 2: Core Backend Development
    - Implement business logic in views
- Sprint 3: Frontend Foundation
    - Create homepage
    - Allow users to filter through class listing (department/f credits)
    - Ability to return to homepage a research a different class
- Sprint 4: Visualization & Predictions
    - Create algorithm for prediction
    - display info in templates align with course info
- Sprint 5: Features & Integration
    - Enhance prediction accuracy
    - Implement advanced filtering options
- Sprint 6: Refinement & Deployment
    - Bug fixes and performance optimization
    - Documentation completion
    - Final deployment preparation