Mouryan's Script (MVC model and DFD Level 1)

Slide 1: Introduction to UMI's Architecture
"Hello everyone, I am Mouryan and today we will explore how UMI leverages its MVC structure and DFD components to create a seamless, user-centric experience."

DFD Overview – UMI's Task and Data Flow Design

"Let's start by looking at how UMI handles data flow. At the center of UMI's design is the user, who inputs tasks, deadlines, and study preferences. In return, they receive reminders and progress updates to stay on track.
The Task Manager Module processes these inputs, storing tasks securely in the Notion API, which is our primary data storage. This API ensures that tasks and schedules are always up-to-date, sending completion data as tasks are finished.
The Task Manager also coordinates with the Calendar Sync Module to integrate user tasks with their preferred calendar app, providing real-time sync updates to ensure accuracy.
To keep users motivated, the Progress and Feedback Module tracks completed tasks and provides summaries, showing them how much they've achieved.
The Pomodoro Timer Module is another key component, supporting timed study sessions. It can play custom playlists through the Spotify API for a personalized study environment."

Model – UMI's Memory
"Now, let's dive into UMI's architecture, beginning with the Model, which you can think of as UMI's memory. The Model securely stores all user data, from tasks and schedules to Pomodoro sessions and playlists. By organizing data into structured collections or tables, it ensures quick retrieval. Temporary data, like an ongoing Pomodoro session, is stored locally, allowing UMI to respond instantly.
In essence, the Model makes sure that all your data is safe, organized, and ready to use whenever you need it."

View – UMI's Interface
"Next, we have the View. Built with HTML, CSS, JavaScript, and React, the View is what users see and interact with. This interface isn't just functional—it's designed to be enjoyable and easy to navigate.
In the View, users can switch between light and dark modes, access their task planner, calendar, and Pomodoro timer, and customize their experience with themes and music preferences. This means the View is not only responsive but also tailored to fit each user's style, making productivity feel effortless."

Controller – UMI's Middleman
"Finally, we have the Controller, which acts as a bridge between the Model and the View. The Controller takes user actions, like adding tasks or starting a Pomodoro session, and updates the Model. Then, it directs the View to display this updated information, so the user is always seeing the latest data.

Additionally, the Controller handles external API interactions. For example, when the user syncs a task to their calendar, the Controller communicates with Google Calendar or Notion, ensuring data accuracy and handling errors or validations. This allows UMI to be fully synchronized and always up-to-date."