WIL PROJECT

Program Name: Full Stack Software Development

Project Code: CPL-5559-FSDS

Week 4

Applicable VLOs or EESs for This Week's Case Study

5. Select, apply, and implement software applications adhering to industry-standard architectural patterns commonly used to develop full-stack software applications.

This Week's Detailed Case Study Information

This morning, your workspace feels more like a stage than a workstation. Three holographic panels float in sync over your desk: one labelled *Model*, one *View*, one *Controller*. They pulse faintly, waiting.

Yelena Ward enters silently, reviewing lines of code projected across her AR lens. She glances up.

"Abstraction matters," she says without preamble. "The difference between a rushed build and a resilient system often comes down to how you separate responsibility."

She gestures toward the three floating panels.

"Model. View. Controller. You've heard it before. Today, you'll live it."

She snaps her fingers, and the panels shift, displaying mock apps, a temperature converter, a to-do list, a tip calculator, and even a minimalist budgeting tool.

"Choose something simple, but build it *clean*. Your logic lives in the model. Your controls should never talk directly to the view. And your view? That's the translator. It shows only what it's told."

She turns to leave, then adds one last note.

"Anyone can make something that works. Today, make something that stays organized when it grows."

WIL PROJECT

Program Name: Full Stack Software Development

Project Code: CPL-5559-FSDS

Deliverables for This Week's Case Study

- Design and build a mini app using the MVC pattern.
    - Choose one of the following (or you can propose your own):
        - A basic calculator (add, subtract, multiply, divide)
        - A currency or temperature converter
        - A simple to-do list
        - A single-category budgeting tool
    - Implement the app using HTML, CSS, and JavaScript, or a front-end framework that naturally supports MVC
- Your code should reflect the MVC structure:
    - Model: Manages the data and business logic
    - View: Handles layout and UI rendering
    - Controller: Connects input/actions to logic and updates the view accordingly
- Submit a folder or link to your organized codebase.
    - Share your project via GitHub, CodeSandbox, or a similar platform
    - Make sure each part of MVC is separated and clearly labelled or structured in folders/files