📅 Calendar Application

Modern Event Management System

ASE 420 - Software Engineering

Presented by: Joshua Day

Date: November 19, 2025

🎯 Project Overview

Feature-rich calendar with dual views

- Monthly grid & Weekly column views
- Event management with color coding
- MongoDB cloud database integration
- Desktop-optimized design
- Live clock with 12-hour format

Used: HTML5, CSS3, JavaScript ES6+, Bootstrap 5, MongoDB

📊 Sprint Timeline

Sprint 1: Sept 15 - Oct 13 (4 weeks)

- Foundation & Core Features

Sprint 2: Oct 15 - Nov 23 (6 weeks)

- Enhancement & Polish

Sprint 1 Weekly Progress

Week 1 (Sept 14-20): Set up project structure and built monthly calendar grid

Week 2 (Sept 21-28): Implemented weekly view with vertical day columns

Week 3 (Sept 28-Oct 4): Created event/task creation and management system

Week 4 (Oct 5-11): Added LocalStorage for persistent data storage

🚀 Sprint 1: What Went Well ✅

- Calendar grid math was straightforward
- Bootstrap made UI styling fast and easy
- LocalStorage was simple to implement
- HTML/CSS came naturally
- Event data structure design was clear

🔴 Sprint 1: Challenges ⚠️

- original design
- Current day feature
- Positioning events in weekly view was tricky
- Getting days to line up correct
- Making events clickable/editable

Sprint 2 Weekly Progress

Week 6 (Oct 12-18): Added comprehensive code comments throughout codebase

Week 7 (Oct 19-25): Implemented live clock with real-time 12-hour display

Week 8 (Oct 26-Nov 1): Removed duplicate CSS and cleaned up UI animations

Week 9 (Nov 2-8): Major refactoring for performance improvements

Week 10 (Nov 9-15): Reorganized file structure and added final polish

Week 11 (Nov 16-22): Integrated MongoDB for cloud database storage

🎨 Sprint 2: What Went Well ✅

- Live clock was easy to implement with setInterval
- File organization made code easier to find
- Removing duplicate CSS
- Documentation helped to lay out what does what
- UI polish borders andcentering looked more appealing

🔴 Sprint 2: Challenges ⚠️

- MongoDB integration (urrent work in progress)
- Converting 24hr to 12hr format
- Fixing horizontal scroll took multiple attempts
- Getting data to sync and convert for mongo

Design Patterns Used

1. Singleton Pattern

Where: One Calendar instance for entire app

Why: Ensures single source of truth for events and state

2. Module Pattern

Where: Calendar class encapsulates all methods

Why: Keeps data private, organizes related functionality

3. Observer Pattern

Where: Event listeners (click, save, delete buttons)

Why: UI responds automatically to user actions

=Design Patterns Used (cont.)

4. Factory Pattern

Where: `createEventElement()` method

Why: Consistent way to create event DOM elements

5. Strategy Pattern

Where: `switchView()` chooses monthly or weekly rendering

Why: Different algorithms for different view types

6. Repository Pattern

Where: `loadEvents()` and `saveEvents()` methods

Why: Abstracts storage (easy to swap LocalStorage for MongoDB)