Week 5: Frontend Basics

CSI403 Full Stack Development | Lab 4 (8%)

Semester 1/2569

Agenda

The Frontend Stack

HTML - Structure and Content

CSS - Styling and Layout

JavaScript - Interactivity

HTML - Structure

Defines the structure of web pages

- Headings, paragraphs, lists
- Links and images
- Forms and inputs
- Tables and containers

CSS - Styling

Controls visual appearance

- Colors and fonts
- Spacing and borders
- Layout (flexbox, grid)
- Responsive design

JavaScript - Interactivity

Adds dynamic behavior

- Handle user events
- Modify page content
- Make API requests
- Form validation

HTML5 Document

```html
<!DOCTYPE html>
<html lang="th">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width, initial-scale=1.0">
    <title>TaskFlow</title>
    <link rel="stylesheet" href="css/style.css">
</head>
<body>
    <h1>Welcome to TaskFlow</h1>
    <script src="js/main.js"></script>
</body>
</html>
```

HTML Forms

```html
<form id="taskForm">
    <label for="title">Title:</label>
    <input type="text" id="title" name="title" required>

    <label for="priority">Priority:</label>
    <select id="priority" name="priority">
        <option value="low">Low</option>
        <option value="medium">Medium</option>
        <option value="high">High</option>
    </select>

    <button type="submit">Create Task</button>
</form>
```

What is Bootstrap?

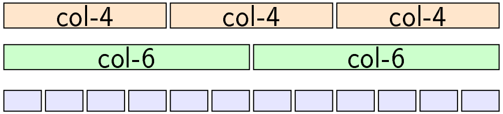
CSS Framework for Responsive Design

- Pre-built CSS classes
- Responsive grid system
- Ready-to-use components
- Mobile-first approach

Include Bootstrap via CDN

```
<head>
    <link href="https://cdn.jsdelivr.net/npm/
        bootstrap@5.3.0/dist/css/bootstrap.min.css"
        rel="stylesheet">
</head>
<body>
    <!-- Content -->
    <script src="https://cdn.jsdelivr.net/npm/
        bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js">
    </script>
</body>
```

Bootstrap Grid: 12 Columns



Columns must add up to 12

Grid System Code

```
<div class="container">
    <div class="row">
        <div class="col-6">Left Half</div>
        <div class="col-6">Right Half</div>
    </div>

    <div class="row">
        <div class="col-4">Column 1</div>
        <div class="col-4">Column 2</div>
        <div class="col-4">Column 3</div>
    </div>
</div>
```

Responsive Breakpoints

Class	**Width**	**Device**
col-	< 576px	Phone
col-sm-	>= 576px	Phone landscape
col-md-	>= 768px	Tablet
col-lg-	>= 992px	Laptop
col-xl-	>= 1200px	Desktop

Bootstrap Card

```html
<div class="card">
    <div class="card-header">Task Title</div>
    <div class="card-body">
        <p class="card-text">Description...</p>
        <span class="badge bg-warning">Pending</span>
    </div>
    <div class="card-footer">
        <button class="btn btn-primary btn-sm">Edit</button>
        <button class="btn btn-danger btn-sm">Delete</button>
    </div>
</div>
```

Bootstrap Form

```html
<form>
    <div class="mb-3">
        <label class="form-label">Title</label>
        <input type="text" class="form-control">
    </div>
    <div class="mb-3">
        <label class="form-label">Status</label>
        <select class="form-select">
            <option value="pending">Pending</option>
            <option value="done">Done</option>
        </select>
    </div>
    <button class="btn btn-primary">Save</button>
</form>
```

Variables

```javascript
// Modern JavaScript uses let and const
const appName = "TaskFlow";  // Cannot reassign
let count = 0;               // Can reassign

// Data types
const title = "My␣Task";     // String
const id = 1;                // Number
const isActive = true;       // Boolean
const items = [1, 2, 3];     // Array
const task = {               // Object
    id: 1,
    title: "Task"
};
```

Functions

```
// Traditional function
function greet(name) {
    return "Hello,␣" + name;
}

// Arrow function (modern)
const greet = (name) => {
    return "Hello,␣" + name;
};

// Short arrow function
const greet = (name) => "Hello,␣" + name;

// Call function
console.log(greet("Student"));
```

DOM Selection

```javascript
// Get element by ID
const form = document.getElementById('taskForm');

// Get element by selector
const button = document.querySelector('.btn-primary');

// Get multiple elements
const cards = document.querySelectorAll('.card');

// Access form inputs
const title = document.getElementById('title').value;
```

Event Handling

```
// Add event listener
const form = document.getElementById('taskForm');

form.addEventListener('submit', function(event) {
    event.preventDefault();  // Stop form from submitting

    const title = document.getElementById('title').value;
    console.log('Creating␣task:', title);

    // Call API or process data
    createTask(title);
});
```

Create DOM Elements

```javascript
function addTaskToList(task) {
    // Create new element
    const li = document.createElement('li');
    li.className = 'list-group-item';
    li.textContent = task.title;

    // Add to page
    const taskList = document.getElementById('taskList');
    taskList.appendChild(li);
}
```

What is Fetch API?

Modern way to make HTTP requests

- Built into browsers (no library needed)
- Promise-based (async/await)
- Replaces older XMLHttpRequest
- Works with JSON APIs

GET Request

```
async function loadTasks() {
    const response = await fetch('/api/tasks');

    if (!response.ok) {
        throw new Error('Failed␣to␣load␣tasks');
    }

    const tasks = await response.json();
    console.log(tasks);

    // Render tasks to page
    tasks.forEach(task => addTaskToList(task));
}
```

POST Request

```
async function createTask(taskData) {
    const response = await fetch('/api/tasks', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(taskData)
    });

    if (response.ok) {
        const newTask = await response.json();
        addTaskToList(newTask);
    }
}
```

DELETE Request

```
async function deleteTask(taskId) {
    const response = await fetch('/api/tasks/' + taskId, {
        method: 'DELETE'
    });

    if (response.ok) {
        // Remove from page
        const element = document.getElementById('task-' + taskId);
        element.remove();
    }
}
```

Lab 4: Frontend Basics (8%)

Requirements:

1. Create dashboard.html with task statistics
2. Create tasks.html with task list
3. Use Bootstrap 5 grid and components
4. JavaScript for form handling
5. Fetch API to call backend
6. Modal for create/edit task

Lab 4 Grading Rubric

Criteria	**Points**
HTML Structure	2%
Bootstrap Components	2%
JavaScript + Fetch API	2%
Custom CSS	1.5%
Responsive Design	0.5%
Total	**8%**

Questions?

Build the TaskFlow UI!

Next Week: Jinja2 Templates