NomNomSafe

Sprint 1 – Week 5 Recap

ASE 485 – Capstone Project

Jeff Perdue

Week 5 Focus: Front-End Refactoring

Goal

Improve the **structure and maintainability** of the business-facing MVP **without changing behavior**.

Context

- Sprint 1 plan (PPP): Feature 2 – **Refactor MVP Features (Front-End)**
- Week 4 delivered **Feature Isolation** (MVP vs non-MVP)
- Week 5 builds on that clean surface to:
  - Apply **SOLID** principles
  - Introduce **design patterns** where they help
  - Standardize validation and error handling

Week 5 Objectives

From the Sprint Plan

- Preserve **all** externally observable MVP behavior
- Make validation and error handling **consistent**
- Expose only **stable**, fully refactored MVP functionality
- Apply **good design patterns** (Facade, Strategy, Chain of Responsibility)
- Reduce **duplicate logic** and technical debt in the front-end

Feature 2: Refactor MVP Features (Front-End)

Active MVP Surface (Business Side)

- Sign in / sign out
- Single menu management (view menus; business-facing dashboard)
- Menu items CRUD (add, edit, delete)
- Business CRUD (edit business info)
- Account CRUD (edit email / password)
- Setup wizard / onboarding flow

> Week 5 keeps these flows **intact**, but changes **how** the front-end code is structured.

Task 1 – API Facade

What I Implemented

- New API layer: `client/src/api/`
  - `config.js` – Single place for base URL (`http://localhost:5000`)
  - `client.js` – Thin HTTP wrapper (JSON parsing, error normalization)
  - `index.js` – Facade with domain methods:
    - `auth` – signIn, signUp, logout, editLogin, setBusiness
    - `businesses` – getById, create, update
    - `menuItems` – getMenuForBusiness, getByMenuId, updateItem, addMenuItem
    - `allergens` – list

Why

Task 2 – Validation & Feedback Consistency

What Changed

- Extended `utils/formValidation.js`:
  - `isRequired`, `minLength`, `isValidUrl`
  - Hardened `validateEmail` and `validatePassword`
- Replaced all MVP `alert()` calls with:
  - `ErrorMessage` (errors)
  - `ConfirmationMessage` (success)
- Applied consistently in:
  - `MenuItemsPage`, `AddMenuItem`, `MenuItemPanel`
  - `SetUp` (business name conflicts)

Why

Task 3 – ProtectedRoute Refactor

Before

- One component with a **single block of conditionals** handling:
  - Auth vs non-auth
  - Admin vs non-admin
  - Setup vs non-setup
  - Business association

After

- `ProtectedRoute.jsx` uses a **Chain of Responsibility**:
  - Small guard functions:
    - `allowChooseBusinessGuard`
    - `preventSignedInAccessingAuthGuard`
    - `adminGuard`
    - `setupGuard`
    - `businessAssociationGuard`
  - A simple `runLoggedInGuards(context)` evaluates them in order

Result

- Same routing behavior, **cleaner** and **testable** guard logic

Task 4 – AddMenuItem & Allergens

What Changed

- `AddMenuItem` refactored to honor **Single Responsibility**:
  - Panel handles one item’s UI and local state
  - Parent manages list of panels and “Save All”
- Allergen data:
  - `allergenCache.js` now uses the API facade (`api.allergens.list()`)
  - `AddMenuItem` and `MenuItemPanel` share this single source of truth
- Validation & feedback:
  - “Name required” and API failures now surface via shared `ErrorMessage` / `ConfirmationMessage`

Why

- Removes duplicated allergen logic
- Makes it easier to extend or change allergen behavior later

Task 5 – Consistency Pass on MVP Screens

Updated to Use Facade + Shared Patterns

- `EditBusinessInfo` – Front-end business GET/PUT via `api.businesses`
- `SetUp` – Business creation + update via `api.businesses`
- `ChooseBusiness` – Business listing and `set-business` via `api.businesses` / `api.auth`

Result

- All MVP front-end code now:\n- Uses the **API facade** for HTTP\n- Uses shared **validation and feedback** components\n- Is aligned with the same design principles across the board

Restoring Menu / Menu Item CRUD Access

Problem

After removing the original default "master menu", **new businesses** had: No menu record in the backend yet

- No obvious way to reach menu item CRUD from the dashboard

Solution

- `MenuDashboard`: When a business has **no menus**, show: - "No menu exists for this business yet."
  - A “**Set up your menu**” button → `/menuitems`\n- `MenuItemsPage`:\n - If no menu exists, synthesize a **deterministic menu id**: `menu_${businessId}`\n - Save it as `currentMenuId` in `localStorage`\n - Use that id for menu item CRUD via existing endpoints

Design Patterns in Practice

Facade

- `client/src/api/`
- Simplifies HTTP access and centralizes endpoint contracts

Strategy

- `utils/formValidation.js`
- Validators as reusable strategies (email, password, required, URL, etc.)

Chain of Responsibility

- `ProtectedRoute.jsx`
- Ordered guard functions manage complex routing decisions without a giant `if` block

Outcomes vs Sprint 1 Plan

Sprint 1 – Feature 2 (Front-End Refactor)

- ✅ MVP behavior preserved (no feature regressions)
- ✅ Front-end HTTP access centralized behind an API facade
- ✅ Validation and error handling standardized
- ✅ Route guards refactored into a maintainable pattern
- ✅ Menu CRUD access restored for businesses without preexisting menus

This completes the Week 5 milestone for **Feature 2: Refactor MVP Features (Front-End)** and positions us for future work on **search/filter/sort** and **onboarding improvements** in Weeks 6 and 7.

Sprint 1 Completion Metrics

- Features completed so far (Jeff, Sprint 1 Week 5): **2 / 4**
- Completed: Feature 1 – Archive Non-MVP Features
- Completed: Feature 2 – Refactor MVP Features (Front-End)
- Actor–goal requirements completed so far: **10 / 22**
- Completed this sprint: R2.1–R2.5 (added to R1.1–R1.5 from Week 4)
- Burndown rate: **45%**

Looking Ahead

Next in Sprint 1 (From PPP)

- **Week 6:** Expanded Search, Filter, and Sort
  - Improve item discovery for business users
- **Week 7:** Improved Business Onboarding
  - Clarify required data and validation paths

The Week 5 refactor gives us a **cleaner, more reliable front-end foundation** to build these features on top of.