CSC 510 Project 1A1 - Fall 2025

Group 28 - Dhananjay Raghu, Ishaan Patel, Shuba Kalyanasundaram

Stakeholders

- **Primary users**
  - Customer (places orders, views totals, picks up.)
  - Staff - Barista/Fulfillment (manages inventory, views & fulfills orders.)
  - Administrator (manages users, sets tax rate.)
- **Upper management**
  - Café operations manager (throughput, queue health)
  - Inventory manager (stock accuracy)
- **Project team**
  - Developers
  - QA / Test Lead (manages acceptance tests, ≥70% backend coverage)
  - CI/CD Owner / DevOps (manages GitHub Actions, coverage, badges)
  - Security & Compliance Lead (manages accessibility, privacy/compliance, and security review)

Stakeholder biases

1. Staff and Customers would not see eye to eye. Customers want their service done as quickly as possible. Staff would like to see their workload reduced, made more efficient with software carrying the load.
2. Administrators and Upper management focus on higher profit margins and increased revenue. Customers focus on lower costs for the same output.
3. Administrators and Staff would like the software to be released and maintained as quickly as possible. Developers would like their workload to be reduced and typically prioritize quality and maintainability of the code at the cost of time.
4. Administrators will typically lean towards more features and customizability of the software while Developers are more concerned on the core business requirements of the product while keeping it simple.
5. Upper management will try to find ways to maximize profits using this software (for example, adding a "*Add a pastry for $2 more*" option). Customers typically do not like to see these options and would like a simpler experience.

Prompt Crafting: Zero Shot Prompting vs Careful Prompting

1. Started with zero-shot prompting where a brief project overview was provided and the model was asked to "create a list of stakeholders for this product." The model responded quickly with several stakeholders, though about half proved irrelevant to the specific context.

2. Next, the stakeholder list was refined using internally selected candidates and useful items from the earlier output. The curated set was then fed back to the LLM with careful prompting, constrained to that list and requesting specific, scenario-based bias examples; this produced more targeted, relevant outputs and clarified edge cases.
3. Overall, careful prompting yielded information that was more relevant and immediately usable for the project's needs. Although additional effort was required to frame constraints and structure the request, the outcomes were preferable. By contrast, zero-shot prompting was faster to issue but slower to evaluate, since extra time was spent sifting less precise responses.
4. **Takeaway:**
    - **Zero-shot prompting:** Give the model a task with minimal context and no examples/constraints. This is fast to type, but broad and sometimes off-target. Use zero-shot for quick exploration/brainstorming
    - **Careful prompting:** Specify goals, constraints, format, and sometimes examples. This takes more setup, but yields precise, consistent outputs. Use careful prompting when you need accuracy, relevance, and consistency.

Use Cases

1. **Create Item/Recipe (Staff)**
   Preconditions: Staff member is logged in and authorized to edit the menu. All required data to create item/recipe (eg. inventory location, units) are present in the system.
   Main flow:
    - Staff selects *Create Item / Create Recipe*
    - System shows the form to create recipe (eg. name, duration, ingredients, relevant media, price)
    - Staff completes required fields and adds required ingredient
    - Staff adds any optional fields such as images of the dish/item.
    - Staff hits *Preview* and is shown the preview of the recipe to be added to the system
    - Staff confirms details and Submits
    - System confirms to staff with success message

   Subflows:
    - If ingredient is not present in system and staff needs it to finish creating recipe, they may be able to create it inline and attach it in the current running workflow
    - When uploading images, system will validate size, file type and store image

   Alternate flows:
    - Invalid or empty required fields: do not continue, prompt user to re-enter
    - Failed to upload image: do not continue, prompt to re-enter
    - User role/Permissions denied: return 403 Forbidden

2. **Add Inventory (Staff)**
   Preconditions: Staff member is logged in and authorized to edit the menu. All required data to create item/recipe (eg. inventory location, units) are present in the system.
   Main flow:
   - Staff goes to Inventory section and selects *Add*
   - Staff selects an existing Inventory unit (or creates new one) and enters details (eg. quantity, location, price, expiration, etc)
   - Staff confirms inventory form
   - System creates a receipt of newly created inventory and displays to staff
   - System returns success code and updates inventory in DB

   Subflows:
   - Create inventory unit: Create metadata for the inventory slot to use
   - Unit of measurement creation: Add UOM enum and store in system (eg. in, lb, gal)

   Alternate flows:
   - Invalid or empty required fields (including negative quantities): do not continue, prompt user to re-enter
   - Invalid Inventory unit or UoM: 400 Bad Request
   - User role/Permissions denied: return 403 Forbidden

3. **Purchase Item/Recipe (Customer)**
   Preconditions: Item/Recipe has been created and submitted. Item is mapped to inventory and there is one or
   more available.
   Main flow:
   - Customer selects the item and chooses modifications, quantity, and time window for pickup
   - Customer checks out their items
   - System calculates price, tax, and displays order summary for user
   - Customer inputs payment information and confirms order
   - System process payment and creates order record, displaying success
   - System sends message/event of order to staff
   - System sends customer ETA and relevant information

   Subflows:
   - Apply discount: validate code and recalculate before payment
   - Pay-at-store: skip payment authorization, prompting to staff that the customer will pay in cash/at store

   Alternate flows:
   - Inventory insufficient: show substitution or suggest later pickup
   - Payment failure: prompt retry or alternate payment

4. **Create/Edit Staff User (Admin)**
   Preconditions: Admin has been authenticated and has the proper user roles.
   Main flow:
   - Admin opens Users page and clicks on *Create/Edit Staff*
   - If Create: Admin fills in required details (name, email, phone number, user role, position, username, etc)
   - If Edit: Admin selects a staff user from the dropdown and updates the fields on the Edit User pop up
   - System records the entry and updates users in DB

   Subflows:
   - Password reset: Admin is prompted to confirm whether staff's password should be reset. Once confirmed, an OTP email will be sent to staff members.

   Alternate flows:
   - Invalid role/station: 400 Bad Request
   - Attempt to delete last admin: 403 Forbidden
   - Authorization failure: 403 Forbidden for non-admin attempts.

5. **Create/Edit/Delete Customer User (Admin)**
   Preconditions: Admin is authenticated and authorized to manage customers. Required fields and validation rules are configured.
   Main Flow:
   - Admin selects create customer
   - Enter profile data
   - System validates inputs
   - System checks the uniqueness constraints
   - System creates customer record
   - System shows confirmation and the profile of customer

   Subflows:
   - Display empty form set to customer
   - Enter data
   - Validate and ensure correct formats
   - Check user/email is unique
   - Create a persistent record, hash the password and set timestamps.
   - Show success.

   Alternate flows:
   - To edit a customer, the admin finds the existing customer, updates the data, the system validates the input, and then updates the record before showing the updated profile.

- The admin finds the customer, confirms the action, and the system marks the account as inactive before returning to the list view. Validation Error: The input fails validation, so the system highlights errors and the admin corrects them before retrying.
- Cancels operation nothing is saved.

6. **Place Order with Multiple Items + Tip (Customer)**

Preconditions: Menu item and inventory exist, tax rate and tip options are configured. The payment service is online and reachable.

Main Flow:

- Customer looks at items
- Customer adds items to cart
- System computes cart subtotal tax and tip
- Customer chooses tip amount
- Customer goes to checkout
- Customer enters details for payment
- System authorizes payment
- System creates order and reserves the inventory and then pushes to staff queue.
- System confirms to customer

Subflows:

- Find items, search and browse catalog
- Choose modifiers and options for item
- Find the totals including tax and tip
- Customer picks the preset tip or custom tip
- Customer enters a pickup name time
- Provide payment method
- Save order send order to queue and show on screen confirmation

Alternate Flows:

- If the customer is signed in, saved profile and payment information are used. If the customer is a guest, they provide only the minimum required information.
- The payment fails, so the system shows an error and allows the customer to retry or use a different method. Out-of-Stock Item: At checkout, the system flags an unavailable item and either suggests a substitution or asks the customer to remove it.
- Before payment, the customer adjusts the tip amount, and the system immediately updates the total.
- The customer cancels the order before payment, so no order is created and the cart is either discarded or saved temporarily.

7. **View Orders (Staff)**
   Preconditions: Staff is authenticated as staff and orders exist in the system.
   Main Flow:
   - Staff opens the orders dashboard
   - System shows orders by status, is it incoming, in progress ready completed etc.
   - Staff applies filters or sorts
   - Staff selects an order to view the details
   - Staff updates the order status
   - When an order is ready system will notify customer
   - Order transitions to completed once its handed to customer

   Subflows:
   - Default view for staff
   - Show order status focus on overdue orders
   - Filter by attributes of orders
   - Expand details on items within the order
   - Transition the order status
   - Notify customers on ready.

   Alternate Flows:
   - The system shows an empty state if no orders are available.
   - If the connection is lost, the system displays an offline banner and refreshes the queue once the connection is restored.
   - Staff assigned to specific stations see only the orders relevant to their role.
   - The system encounters a load error, so it displays a retry option and logs the issue.
   - If the order is already fulfilled or canceled, the system shows a conflict message and prompts the staff to refresh the queue.

8. **Fulfill Order (Staff)**
   Preconditions: Authenticated as staff, order exists and is OPEN, items are ready (ops) and services are reachable.
   Main flow:
   - Open Orders Queue → see OPEN orders
   - Select order → Fulfill
   - System validates role/state → sets FULFILLED, writes fulfilledBy/fulfilledAt (atomic)
   - Emit status change (websocket/poll) → pickup display shows Ready
   - Return 200 OK; order leaves OPEN list (or appears under Fulfilled)

   Subflows :
   - Confirm action (lightweight)
   - Audit trail (fulfilledBy, fulfilledAt, prior status)
   - Notify via event/poll flag

- Idempotent: repeat clicks don't double-fulfill

Alternate flows:

- No open orders: show empty state
- Stale state: already fulfilled/cancelled → 409 (refresh UI, disable Fulfill)
- Auth failure: not staff → 403
- Invalid request: bad/missing ID → 400, reselect and retry.

9. Order Pickup Notification (Customer)

Preconditions: Order exists, identified by login or pickup code, staff has FULFILLED the order, pickup screen (public board or "My Orders") and connectivity available.

Main flow:

- Customer opens pickup screen / "My Orders."
- System subscribes (websocket) or polls for status.
- Fulfillment event detected → UI shows Ready for Pickup with code/location.
- Customer taps Got it; entry auto-hides after a short TTL.

Subflows:

- Privacy masking: show code/initials, not names or item details on public boards.
- Accessibility: live regions, keyboard focus, adequate contrast.
- If a customer has more than one order, show each with its own code/status and update them independently.

Alternate flows:

- Wrong code: show not found / retry.
- Cancelled: show Cancelled with help cue.
- Notification/display failure: fallback to polling; show small retry banner.
- Duplicate/out-of-order events: treat idempotently; keep latest status.
- Display offline: personal "My Orders" still updates; staff can call the code.

10. Set Tax Rate (Admin)

Preconditions: Authenticated as admin, settings UI/API available, current tax rate is retrievable.

Main flow:

- Admin opens Settings and selects Tax Rate.
- Enter a new rate and click Save.
- System validates input (numeric, 0–100%, precision/format).
- Persist change atomically, set updatedBy/updatedAt and new version.
- Invalidate caches; subsequent new orders use the new rate; system returns 200 OK and shows confirmation.

Subflows:

- Audit & versioning: Record updatedBy, updatedAt, previous value; keep a change log.

- Precision & rounding: Enforce allowed decimals (e.g., two); document rounding rules for order totals.
- Effective timing: Change applies immediately to new orders only; existing orders are not recalculated.

Alternative flows:

- Authorization failure: Non-admin throws 403 Forbidden.
- Validation error: Non-numeric, out of range, too many decimals throws 422 Unprocessable with inline hint.