1. Stakeholders

1.1. Stakeholder List

Stakeholder	**Group**
Customers (students, faculty)	Users (direct day-to-day interaction)
WolfCafe Staff	Users
Administrators/Managers	Users, Domain Experts
Accessibility Users	Users, Regulators/Legal (indirect)
WolfCafe Business Owner(s)	Clients/Sponsors
University (dining services)	Clients/Sponsors, Indirect
CTOs/Product Owners (instructors)	Clients/Sponsors, Domain Experts
App Development Team	Developers/Engineers
Future Developers	Developers/Engineers, Indirect
IT Support/Operations	Developers/Engineers
Regulatory Bodies (food, IT)	Regulators/Legal
Accessibility Standards Bodies	Regulators/Legal
Competitors/Alternative Vendors	Indirect/Negative

1.2. Stakeholders - Power Interest Grid Classification

- Manage Closely (High Power, High Interest)
  - They shape requirements and must be actively engaged.
  - WolfCafe Business Owner(s) → They directly own the café, control budget, and care deeply about system success.
  - CTOs/Product Owners (instructors) → They define the system’s scope and grade outcomes.
  - Administrators/Managers → Directly run operations, have high interest, and significant influence on what features staff need
- Keep Satisfied (High Power, Low Interest)
  - They have influence but may not care about day-to-day details. Keep them happy but don’t overload with detail.
  - University (dining services, institutional sponsor) → High power (funding, reputation, permissions) but lower daily interest in system mechanics.

    - Regulatory Bodies (food safety, IT compliance) → Can block the system if noncompliant but aren't invested in features.
- Keep Informed (Low Power, High Interest)
    - They care a lot about the system, but have less ability to change requirements. Need regular updates and user-focused design.
    - Customers (students, faculty, staff) → Their satisfaction drives system success, but they don't control development.
    - WolfCafe Staff (baristas, cashiers, food prep) → Very invested in usability (they use it daily) but have little formal power.
    - Accessibility Users (subset of customers/staff) → Their interest is high (they depend on accessibility) but power is indirect (via standards bodies).
- Monitor (Low Power, Low Interest)
    - Peripheral stakeholders, monitor with minimal effort.
    - Future Developers → They care later, but not now.
    - IT Support/Operations → Only relevant once deployed, so low current interest.
    - Accessibility Standards Bodies → Set requirements but day-to-day involvement is minimal.
    - Competitors/Alternative Vendors → Indirectly affected, but no direct say in the system.

1.3. Stakeholder Biases

- Users
    - Customers (students, faculty, staff ordering food)
      → Want fast service, affordable prices, convenience, minimal barriers to ordering.
    - WolfCafe Staff (baristas, prep, cashiers)
      → Want organized, accurate, manageable workflow, with a UI that is fast and simple under pressure.
    - Administrators/Managers (store/shift leads)
      → Want smooth operations, accurate tracking, maximum revenue, and oversight of staff and sales.
    - Accessibility Users (subset of customers/staff)
      → Want accessible, readable, non-flashy UIs, compliant with ADA/WCAG.
- Clients/Sponsors
    - WolfCafe Business Owner(s)
      → Want profitability, customer satisfaction, long-term adoption.
    - University (Dining Services)
      → Want good reputation, alignment with campus systems, compliance.
    - CTOs/Product Owners (instructors)
      → Want feature completeness, adherence to good engineering practices, demonstrable learning outcomes.
- Domain Experts
    - (Same as above: Admins, Instructors → bring deep knowledge of operations/requirements.)
- Developers/Engineers

    - Dev Team
      → Want to experiment with features/tech, build impressive UI/UX, ship a working product that looks modern.
    - Future Developers
      → Want clear documentation, maintainable codebase, no excessive technical debt.
    - IT Support/Operations
      → Want reliability, ease of deployment, minimal maintenance overhead.
- Regulators/Legal
    - Regulatory Bodies (food safety, data security, IT policy)
      → Want compliance with health codes, security policies, data handling rules.
    - Accessibility Standards Bodies (ADA, WCAG)
      → Want accessible interfaces, no exclusion of disabled users.
- Indirect/Negative
    - Competitors (other food vendors)
      → Don't want WolfCafe to succeed → may highlight flaws.
    - Future Developers (inherit system)
      → High interest later, frustrated if poor design choices are made now.

1.4. Stakeholder Clashes

- Customers vs. Staff
    - Conflict: Customers want fast service; staff want manageable workflow and fewer errors.
    - Example: Customers ordering highly customized drinks slows staff down and clutters UI.
- Customers vs. Administrators/Business Owners
    - Conflict: Customers want affordability and minimal logins; admins/owners want higher revenue, loyalty tracking, and strong security.
    - Example: Admins push loyalty programs requiring logins; customers prefer guest checkout.
- Staff vs. Dev Team / IT Ops
    - Conflict: Staff want a simple, reliable UI; dev team may build flashy features; IT Ops want stability.
    - Example: A "modern" animated order board looks cool but crashes under heavy load, frustrating staff.
- Accessibility Users vs. Dev Team
    - Conflict: Dev team favors trendy UI (animations, colors); accessibility users need readable, non-flashy design.
    - Example: High-motion effects impair navigation for visually impaired users.
- University/Regulators vs. Customers
    - Conflict: University and regulators want strict compliance (auth, food safety logs, data protection). Customers want speed and convenience.
    - Example: Multi-step login slows down ordering → customers frustrated.

2. Comments on Prompt Crafting

2.1. Zero-Shot Prompting

- Definition: This method involves providing an LLM with a simple, direct instruction without any specific context, examples, or formatting rules. For example: "Write 10 use cases for a food delivery system."
- Pros: It's fast, straightforward, and requires minimal upfront effort.
- Cons: The output is often generic, lacks key details, and is poorly formatted. It's highly unlikely to meet the specific requirements of the WolfCafe project, such as including the required tip percentages, handling user roles, or reflecting the existing CoffeeMaker functionality.

2.2. Careful Prompting

- Definition: This method involves giving the LLM detailed instructions and specific formatting requirements, integrating prompting skills (e.g. CoT, Prompt Chaining) and one or more high-quality examples. It's like providing a detailed script for the LLM to follow. For example: "Please write 10 use cases for the WolfCafe system. Each use case must include: Use Case Name, Preconditions, Main Flow, Subflows, and Alternative Flows. Here is an example: [a use case example]."
- Pros:
  - High Precision: The output is highly relevant to the problem and follows the specified constraints, such as differentiating between a staff member's permissions and an administrator's permissions.
  - Consistency: All generated use cases will have the same, well-defined structure, making them easy to integrate into a formal requirements document.
  - Better Quality: Providing a good example "teaches" the LLM how to generate high-quality, academically sound use cases that include crucial elements and a professional tone. This directly supports the learning outcomes of your course.
- Cons:
  - Time and Effort: It requires a significant amount of upfront effort to fully understand the project's requirements, analyze the problem, and manually craft a detailed prompt. This time could be spent on other project tasks.
  - Increased Cognitive Load: The user must be highly familiar with the project's nuances to craft an effective prompt. For a complex system like WolfCafe, this means understanding the roles of Admin, Staff, and Customer, as well as the specific features to be implemented.
  - Risk of Garbage In, Garbage Out: If your example or instructions contain errors or are poorly structured, the LLM may replicate those flaws, producing incorrect or unusable output. This can lead to time spent debugging your prompt rather than the generated content.
  - Increased Cost**:** LLM APIs typically charge based on the number of "tokens" (pieces of words) in both prompt and the model's response. A long, detailed prompt with examples uses more input tokens, and the structured, lengthy

response it generates uses more output tokens. This makes it a more expensive method.

3. Use Cases

3.1. UC1 Customer Places an Order

This use case describes how a Customer browses the WolfCafe menu, builds a cart, and completes the checkout process, resulting in a confirmed order that is passed to Staff for fulfillment

3.1.1. Preconditions

- The Customer is authenticated in the WolfCafe system (via account or guest login, if allowed).
- At least one item has already been added to the shopping cart.
- The payment gateway and order services are online.

3.1.2. Main Flow

The Customer proceeds to checkout from their cart. On the summary screen, they can adjust items [Edit Cart], apply a promotional code [Apply Coupon], add a tip, and select a payment method before submitting. The system validates the order, triggering [Payment Declined] for failed transactions or [Item Out of Stock] if an item's availability changes. A successful submission generates a confirmed order and a unique order number. If the customer is inactive for too long, a [Session Timeout] occurs, clearing the cart.

3.1.3. Subflows

- **[Edit Cart]**: The Customer can add, remove, or update item quantities. If any selected item is unavailable, [Item Out of Stock] is triggered, and the Customer must confirm the revised cart.
- **[Apply Coupon]**: The Customer enters a promotional code. The system validates the code and applies a discount. Invalid or expired codes generate an inline error, allowing retry without disrupting checkout.

3.1.4. Alternative Flows

- **[Payment Declined]:** Triggered at checkout submission if the payment provider rejects the transaction. The system notifies the Customer and returns them to payment selection.
- **[Item Out of Stock]:** Triggered when an item is unavailable at checkout or during [Edit Cart]. The system removes the item, updates totals, and prompts the Customer to confirm the updated order.
- **[Session Timeout]:** Triggered if the Customer remains inactive for a configured time. The system logs the user out and clears the cart, requiring the Customer to log back in and rebuild their order.

3.2. UC2 Fulfill Customer Order

This use case describes how Staff process and complete a customer order once it has been placed.

3.2.1. Preconditions

- The staff member is logged into the WolfCafe system with Staff privileges.
- At least one order exists in “Pending” status.
- The required ingredients and equipment are available in the café.

3.2.2. Main Flow

A staff member claims a pending order [Claim Order] from the fulfillment queue, which fails if it is [Already Claimed]. They [View Order Details] and prepare the items, which automatically triggers [Adjust Inventory]. The system issues a [Low Stock Alert] for depleted ingredients. To handle issues, staff may [Substitute Item] or [Contact Customer] to clarify special instructions [Verify Special Instructions]. If a resolution cannot be reached, the order is escalated for [Order Cancellation]. Once complete, the order is marked "Ready for Pickup," initiating UC8. An Admin can reverse this status if it was set in error [Incorrect Fulfillment]. A failure during customer notification triggers [Order Notification Failure].

3.2.3. Subflows

- **[View Order Details]**: Opens the order to show items, modifiers, and notes. Invalid/canceled orders lead to [Order Cancellation].
- **[Claim Order]**: Staff member locks the order to themselves; if already claimed, see [Already Claimed].
- **[Adjust Inventory]**: Deducts ingredient counts; if insufficient, trigger [Low Stock Alert] or [Substitute Item].
- **[Verify Special Instructions]**: Confirms customer notes (e.g., allergies). If unmet, trigger [Contact Customer] or [Order Cancellation].

3.2.4. Alternative Flows

- **[Already Claimed]**: Another staff member has already claimed this order; system blocks duplicate work.
- **[Low Stock Alert]**: Triggered in [Adjust Inventory]; flags shortages and may allow partial prep or substitution.
- **[Substitute Item]**: Triggered if substitutions are allowed; system records change and updates pricing.
- **[Contact Customer]**: Triggered if instructions cannot be fulfilled; if no resolution, escalate to [Order Cancellation].
- **[Order Cancellation]**: Triggered when the order cannot be completed; system logs reason, updates status, and notifies customer.
- **[Incorrect Fulfillment]**: Triggered if order incorrectly marked ready; Admin resets status.
- **[Order Notification Failure]**: Triggered if notification delivery fails; retries or uses fallback channel.

3.3. UC3 Add Menu Item

This use case describes how an Admin creates a new menu item in the WolfCafe system, including assigning metadata, recipe details, and pricing so that the item becomes available to customers.

3.3.1. Preconditions

- The Admin is authenticated in the system with Admin privileges.
- All required ingredients for the new recipe already exist in the inventory.

3.3.2. Main Flow

An Admin adds a new menu item by navigating to the management interface to [Enter Details] (name, price) and [Define Recipe] (ingredients, quantities). The system validates all data, preventing submission with [Invalid Input] or a [Duplicate Item Name]. The Admin may save the new item, making it available to customers, or [Cancel Creation] to discard all changes.

3.3.3. Subflows

- **[Enter Details]**: The Admin provides the item's name, description, and price. Invalid entries (e.g., empty name or negative price) trigger [Invalid Input].
- **[Define Recipe]**: The Admin selects ingredients and specifies their required quantities. If an invalid or unavailable ingredient is selected, the Admin must adjust before saving.

3.3.4. Alternative Flows

- **[Duplicate Item Name]:** Triggered at save if the entered item name already exists. The system rejects the entry and requires the Admin to provide a unique name, returning to [Enter Details].
- **[Invalid Input]:** Triggered if fields contain invalid values (e.g., blank name, negative price, non-numeric quantity). The system highlights the errors and returns to [Enter Details] or [Define Recipe].
- **[Cancel Creation]:** Triggered if the Admin exits or cancels before saving. The system discards all entered data, and no new item is created.

3.4. UC4 Manage Customer Profile

This use case describes how a Customer updates their account information, including personal details and saved payment methods, to keep their profile current and secure.

3.4.1. Preconditions

- The Customer is authenticated in the WolfCafe system with an active session.

3.4.2. Main Flow

The Customer navigates to their account settings to [Edit Personal Info] or [Manage Payment Methods]. The system validates all changes, preventing an update due to a [Password Mismatch] or other [Invalid Input]. The Customer can save the new information or [Cancel Edit] to revert all modifications.

3.4.3. Subflows

- **[Edit Personal Info]**: Customer edits name, email, or password. Password changes require verification of the current password. If verification fails, [Password Mismatch] is invoked, and the update is blocked.
- **[Manage Payment Methods]**: Customer adds, updates, or deletes saved payment cards. Invalid data formats (e.g., expired card, incorrect number) trigger [Invalid Input], preventing the update until corrected.

3.4.4. Alternative Flows

- **[Invalid Input]**: Triggered when data is improperly formatted (e.g., invalid email address, expired card). The system highlights the issue and requests correction.
- **[Password Mismatch]**: Triggered during a password change when the current password provided does not match the stored record. The system prevents the update until the correct password is entered.
- **[Cancel Edit]**: Triggered when the Customer exits the profile page or clicks cancel before saving. The system discards all changes and retains the last saved profile information.

3.5. UC5 View Order History

This use case describes how a Customer reviews their past orders in the WolfCafe system and, if desired, re-orders items for convenience.

3.5.1. Preconditions

- The Customer is authenticated in the WolfCafe system.
- At least one completed order exists in the system (otherwise see [No Order History]).

3.5.2. Main Flow

The Customer accesses their order history, which displays a message if there is [No Order History]. They can select a past order to [View Order Details] and use the [Re-order] function to add all items to their current cart. The system automatically removes any unavailable items from the cart, triggering [Item Out of Stock] and notifying the customer of the change. A [System Error] is shown if the history cannot be retrieved.

3.5.3. Subflows

- **[View Order Details]**: Displays full details of a past order, including purchased items, quantities, modifiers, and prices.
- **[Re-order]**: Adds all items from the selected past order into the Customer's shopping cart. If an item is no longer available, [Item Out of Stock] is invoked.

3.5.4. Alternative Flows

- **[No Order History]**: Triggered when the Customer navigates to "Order History" but has never placed an order. The system displays a friendly message such as *"You haven't placed any orders yet!"*.
- **[Item Out of Stock]**: Triggered during [Re-order] if one or more items from the historical order are unavailable. The system automatically removes the unavailable items and updates the cart, prompting the Customer to confirm the revised order.
- **[System Error]**: Triggered if the system cannot retrieve order history (e.g., database connection issue). The system displays an error message and suggests the Customer try again later.

3.6. UC6 Staff Manages Inventory

This use case describes how a Staff member manages stock levels of ingredients and supplies within the WolfCafe system.

3.6.1. Preconditions

- The staff member is authenticated with Staff privileges.
- Ingredients and supply items (e.g., milk, espresso beans, cups) already exist in the inventory database.

3.6.2. Main Flow

A staff member uses the inventory interface to [Search/Filter Inventory] and select an item. They [Update Stock] quantities, with the system validating the entry and preventing [Invalid Input]. Saving a change can trigger a [Low Stock Alert] if the quantity falls below a predefined threshold. The staff can also [Cancel Update] to discard any changes.

3.6.3. Subflows

- **[Search/Filter Inventory]**: Staff uses a search bar or category filters to locate the desired inventory item.
- **[Update Stock]**: Staff enters the new quantity, and the system validates it. If the value is non-numeric or negative, [Invalid Input] is invoked.

3.6.4. Alternative Flows

- **[Low Stock Alert]**: Triggered when the saved stock level is below a threshold; the system flags the item for reorder.
- **[Invalid Input]**: Triggered if entered values are invalid; the system rejects the update and prompts correction.
- **[Cancel Update]**: Triggered if the staff member cancels or navigates away before saving; no changes are stored.

3.7. UC7 Generate Sales Report

This use case describes how an Admin generates a sales report to analyze revenue and operational performance.

3.7.1. Preconditions

- The Admin is authenticated with reporting privileges.
- At least one order with sales data exists in the system (otherwise [No Data] applies).

3.7.2. Main Flow

An Admin generates a sales report by specifying a time frame [Choose Time Period] and applying optional [Select Filters]. The system displays the results, or a [No Data] message if no transactions match the criteria. The Admin can then [Export Report] to a file; a failure in this process triggers an [Export Error].

3.7.3. Subflows

- **[Choose Time Period]:** Admin selects predefined ranges (daily, weekly, monthly) or a custom date range.
- **[Select Filters]:** Admin filters results by category, staff member, or payment type.
- **[Export Report]:** Admin exports the generated report to PDF or CSV format.

3.7.4. Alternative Flows

- **[No Data]:** Triggered when no transactions match the selected period or filters.
- **[Export Error]:** Triggered if the export fails due to technical issues. The Admin is prompted to retry or select another format.

3.8. UC8 Notify Customer of Order

This use case describes how the WolfCafe system notifies a Customer when their order is ready for pickup.

3.8.1. Preconditions

- A Customer has successfully placed an order (UC1).
- Staff has marked the order as Ready for Pickup (UC2).

3.8.2. Main Flow

When staff marks an order as ready for pickup, the system automatically triggers a [Send Notification] for the customer to [View Notification]. A [Notification Failure] prompts the system to retry or use a fallback communication channel.

3.8.3. Subflows

- **[Send Notification]**: System composes and queues notification messages.
- **[View Notification]**: Customer receives and views the notification details.

3.8.4. Alternative Flows

- **[Notification Failure]**: Triggered when a delivery channel fails. The system retries and/or uses fallback channels (e.g., switching from SMS to in-app).

3.9. UC9 Provide Customer Feedback

This use case describes how a Customer provides feedback on a completed order.

3.9.1. Preconditions

- The Customer is authenticated in the system.
- The order has been completed and marked as delivered/picked up.

3.9.2. Main Flow

After an order is completed, a customer can [View Order Details] in their history and leave feedback. They must [Enter Rating] (1-5 stars) and can optionally [Enter Comment]. The system rejects an [Empty Submission] if the rating is missing and stores the feedback for an Admin to [View Feedback Report]. The customer can [Cancel Feedback] at any point before submission.

3.9.3. Subflows

- **[Enter Rating]:** Customer selects a 1–5 star rating; missing input triggers [Empty Submission].
- **[Enter Comment]:** Customer adds optional written feedback.
- **[View Feedback Report]:** Admin later reviews aggregated results.

3.9.4. Alternative Flows

- **[Empty Submission]**: Triggered when a rating is missing; the system prompts correction.
- **[Cancel Feedback]**: Triggered if the Customer cancels or exits before submission.

3.10. UC10 Deploy System Update

This use case describes how the Dev Team deploys a new version of the WolfCafe system during a scheduled maintenance window.

3.10.1. Preconditions

- The Dev Team has packaged and tested the release.
- An Admin has approved a maintenance window.

3.10.2. Main Flow

During a maintenance window, the Dev Team deploys a system update. The process automatically applies database changes [Run Migration] and then executes a series of automated checks [Run Tests]. A [Migration Failure] or [Test Failure] triggers an immediate rollback to the previous stable version. An Admin is notified via [Downtime Exceeded] if the deployment surpasses its allotted time.

3.10.3. Subflows

- **[Run Migration]:** Applies database schema/data changes. Errors trigger [Migration Failure].
- **[Run Tests]:** Executes automated regression and smoke tests. Failures trigger [Test Failure].

3.10.4. Alternative / Error Flows

- **[Migration Failure]:** Triggered if migrations fail; rollback initiated, deployment halted.
- **[Test Failure]:** Triggered if automated tests fail; previous version restored.
- **[Downtime Exceeded]:** Triggered if the deployment surpasses the approved maintenance window; Admin notified.

4. Use Case Traceability Matrix

Stakeholder Traceability Matrix 1: Customer & Business Operations

Use Case	Requirement	Customers	Staff	Admins/Managers	University (Deans)
UC1 - Customer Places Order	R1: Customer Ordering & Checkout	✓	✓		
UC2 - Fulfill Customer Order	R2: Order Fulfillment & Inventory Adjustment	✓	✓	✓	
UC3 - Add Menu Item	R3: Menu Management	✓	✓	✓	
UC4 - Manage Customer Profile	R4: Customer Account & Profile	✓			
UC5 - View Order History	R5: Order History and Ordering	✓			
UC6 - Staff Manages Inventory	R6: Inventory Management		✓	✓	
UC7 - Generate Sales Report	R7: Sales Reporting			✓	✓
UC8 - Notify Customer of Order	R8: Customer Notifications	✓	✓		
UC9 - Provide Customer Feedback	R9: Feedback Collection	✓		✓	
UC10 - Deploy System Update	R10: System Deployment & Updates				

This matrix is the first part of the Use Case Traceability Matrix, focusing on stakeholder groups who either directly interact with the system or have a direct impact on business operations. It maps each Use Case (UC) and its corresponding Requirement (R) to key stakeholders like Customers, Staff, Admins/Managers, and the University Dining Services. This approach ensures every system function can be traced to a specific stakeholder, validating that the system's design meets core business needs and helping to manage closely and keep satisfied stakeholders during development.

Stakeholder Traceability Matrix 2: Technology & Support

Use Case	Requirement	Business Owners	Dev Team	IT Support Ops	QA/Testing Team
UC1 - Customer Places Order	R1: Customer Ordering & Checkout				✓
UC2 - Fulfill Customer Order	R2: Order Fulfillment & Inventory Adjustment				✓
UC3 - Add Menu Item	R3: Menu Management				✓
UC4 - Manage Customer Profile	R4: Customer Account & Profile				✓
UC5 - View Order History	R5: Order History and Ordering				✓
UC6 - Staff Manages Inventory	R6: Inventory Management	✓			✓
UC7 - Generate Sales Report	R7: Sales Reporting	✓			✓
UC8 - Notify Customer of Order	R8: Customer Notifications				✓
UC9 - Provide Customer Feedback	R9: Feedback Collection				✓
UC10 - Deploy System Update	R10: System Deployment & Updates		✓	✓	✓

As the second part of the Use Case Traceability Matrix, this table focuses on the technical implementation and support aspects of the software. It links each use case and its requirements to the Business Owners, Dev Team, IT Support/Operations, and QA/Testing Team. This table highlights the groups responsible for the system's development, deployment, maintenance, and quality assurance. By doing so, it ensures that the system's technical architecture and operational work align with business goals. It also helps keep satisfied and keep informed stakeholders understand their roles and involvement, which reduces communication gaps and ensures the delivery of a stable, reliable product.