

Tasks of Features

Recipe Management (CRUD for Recipes)

ID	Specific Task	Type (UI/Logic/DB)	Detail
1.1.1	Create the form/interface to add and edit recipes.	UI/DB	Fields: Name, Description, Yield (servings).
1.1.2	Implement field validation (required fields, input formats).	Logic	Validate input before saving.
1.2.1	Design the Recipes table with relationships to Categories and Ingredients.	DB	Fields: id, name, description, yield, category_id.
1.3.1	Build a dynamic ingredient entry module per recipe.	UI/Logic	Fields: Ingredient, Quantity, Unit. Connects to Ingredient Catalog (F3).
1.4.1	Implement automatic cost-per-serving calculation.	Logic	Uses prices from Ingredient Catalog.
1.5.1	Add a module for step-by-step cooking instructions with estimated time per step.	UI/Logic	Allow reordering steps.

1.6.1	Implement secure upload and storage of recipe images.	Logic/UI	Save image paths in DB, validate image formats.
1.7.1	Build the category management module (CRUD).	UI/DB	Categories: starter, main course, dessert, beverage, etc.
1.8.1	Implement search bar and filters by name, ingredient, and category.	UI/Logic	Combine filters dynamically.
1.9.1	Create internal API endpoints for recipe CRUD operations.	Logic/DB	Endpoints: recipes with full CRUD.

Smart Unit Conversion

ID	Specific Task	Type	Detail
2.1.1	Create Conversions table with metric imperial equivalents.	DB	Fields: ingredient_id, unit_from, unit_to, factor.
2.2.1	Implement automatic conversion engine.	Logic	Supports g <-> oz, ml <-> cup, etc.
2.3.1	Add density table for each ingredient.	DB	Example: Flour \neq Sugar.
2.4.1	Implement dynamic recipe scaling by servings.	Logic/UI	Automatically updates ingredient quantities.

2.5.1	Create an editable conversion table view for admin users.	UI	Display ingredient-specific conversions.
-------	---	----	--

Ingredient Catalog

ID	Specific Task	Type	Detail
3.1.1	Create Ingredients table.	DB	Fields: id, name, base_unit, unit_price, density.
3.2.1	Implement CRUD for ingredients.	UI/Logic	Create, update, delete, and view ingredients.
3.3.1	Create Suppliers table.	DB	Link suppliers and prices.
3.4.1	Develop a module to manually update ingredient prices.	UI/Logic	Inline editing of price fields.
3.5.1	Link ingredients to suppliers.	Logic/DB	Intermediate table Ingredient_Supplier.

Cost Calculation

ID	Specific Task	Type	Detail
4.1.1	Implement recipe cost engine.	Logic	Formula: (Quantity * Unit Price) / Yield.

4.2.1	Calculate profit margin and suggested price.	Logic	$\text{Selling Price} = \text{Cost} + (\text{Cost} * \text{Margin})$.
4.3.1	Display cost breakdown and margin in the UI.	UI	Fields: Total Cost, Cost per Serving, Margin, Suggested Price.
4.4.1	Implement supplier cost comparison feature.	Logic/UI	Show differences when switching suppliers.

Quotations and Budgets

ID	Specific Task	Type	Detail
5.1.1	Create quotation generation form.	UI	Fields: Recipes, Quantity, Taxes, Discounts.
5.2.1	Implement automatic calculation of total, VAT, and discounts.	Logic	Formula: $\text{Total} = \text{Subtotal} + \text{VAT} - \text{Discount}$.
5.3.1	Generate professional PDF export.	Logic	Include logo, client info, date, and prices.
5.4.1	Build quotation management dashboard.	UI/Logic	Status: Pending, Approved, Completed.
5.5.1	Maintain quotation history in database.	DB/Logic	Link with Users and Recipes.

Calendar

ID	Specific Task	Type	Detail
6.1.1	Build interactive calendar view.	UI	Display events, deadlines, and orders.
6.2.1	Create Events table linked to quotations.	DB	Fields: id, event_type, date, description, quotation_id.
6.3.1	Add custom event creation feature.	UI/Logic	Examples: meeting, delivery, restock.
6.4.1	Implement event notification/reminder system.	Logic	Alerts for upcoming deadlines.
6.5.1	Add color-coded event categories.	UI	Example: quotation, delivery, note.

Authentication and User Management

ID	Specific Task	Type	Detail
7.1.1	Integrate Google OAuth login.	Logic/UI	Authenticate via Google API.
7.2.1	Create Users table with roles (Chef/Client).	DB	Fields: id, name, email, role, token.
7.3.1	Implement post-login redirection by role.	Logic	Chef -> Chef Dashboard / Client -> Client Dashboard.

7.4.1	Maintain session persistence.	Logic	Use secure tokens or cookies.
7.5.1	Build user profile view.	UI	Display name, email, role, and logout option.

Client Quotation Request

ID	Specific Task	Type	Detail
8.1.1	Create client quotation request form.	UI	Fields: Event Type, Date, Guests, Location, Budget, Notes.
8.2.1	Validate and save quotation request to DB.	Logic/DB	Default status: Pending.
8.3.1	Notify chef when a new request is received.	Logic	Redirect to Chef Dashboard.
8.4.1	Display client's quotation request history.	UI/Logic	Show status and date.
8.5.1	Create API for accepting/rejecting quotations.	Logic	Chef responds with price, menu, or comments.