מסמך איפיון מערכת — BOM & Recipe System (The Kosher Place, Thailand)

מטרת המסמך: איפיון מלא ומפורט של המערכת הקיימת — כל המסכים, המנועים, מודל הנתונים וה-API — בתוספת **מפת דרכים מתועדפת** (P0/P1/P2) של מה שצריך להוסיף או לתקן.

---

1. סקירה כללית

מערכת לניהול **מתכונים ועצי-מוצר (BOM)** למטבח/מאפייה, עם תמחור אוטומטי, סנכרון חומרי-גלם מ-**Odoo**, וממשק עברית RTL דו-לשוני (עברית/אנגלית).

- **שני סוגי מתכונים:** מתכון **בסיס** (base) ומוצר **סופי** (final). מתכון יכול לשמש כתת-מתכון בתוך מתכון אחר (קינון רב-שכבתי).
- **חומרי גלם** מסונכרנים מ-Odoo; **מתכונים** נוצרים במערכת.
- **תמחור** מבוסס נוסחאות חופשיות (עלות × מכפיל, מע"מ, עיגול) שניתנות לעריכה.
- **מטבע תצוגה:** ฿ (באט תאילנדי).

טכנולוגיות

שכבה	טכנולוגיה
Backend	Node.js + Express, PostgreSQL (pg)
Frontend	React + TypeScript (Vite), Zustand, React Query
אינטגרציה	Odoo v18 דרך XML-RPC
אקסל	ExcelJS (ייבוא/ייצוא)
אימות	JWT + טבלת `users` מקומית

---

2. משתמשים, תפקידים והרשאות

שלושה תפקידים (`users.role`):

- **customer** — צפייה בלבד בספר המתכונים + מחשבון כמויות. רואה מחירים רק אם `can_view_prices = true`.
- **admin** — ניהול מלא: מתכונים, מוצרים, נוסחאות, משתמשים, סנכרון.
- **manager** — על-קבוצה של admin; **בלעדית** רשאי לקדם מתכוני-טסט למתכונים אמיתיים. רואה תמיד מחירים.

אכיפת הרשאות: `requireAdmin` = manager = `requireManager`, admin|manager בלבד. **נראות מחירים נאכפת בצד-שרת** (`pricesMiddleware` חותך שדות מחיר מכל תגובה למי שאינו מורשה — ה-UI לא נסמך עליו).

אימות (POST /api/auth/login): סדר בדיקה — (1) dev-admin bypass (`ALLOW_DEV_LOGIN`, מקבל manager), (2) סיסמה מקומית (scrypt, `users.password_hash`), (3) Odoo XML-RPC. בכל בקשה התפקיד/הסטטוס נקראים **חי** מטבלת `users` (שינוי הרשאה נכנס לתוקף מיד, בלי התחברות מחדש).

---

3. מודל הנתונים (PostgreSQL — `src/db/schema.sql`)

מיגרציות **אידיטיביות בלבד** (`IF NOT EXISTS`), מורצות ע"י `node src/db/migrate.js`.

טבלה	תיאור ושדות עיקריים
items	חומרי גלם **ומתכונים** מאוחדים. `item_type` ∈ {raw_material, recipe}. שדות: `cost_per_kg` (העלות הקנוני `name/name_en/name_he` , `reference` (SKU), `uom` , `raw_cost` (Odoo standard_price), `weight_extracted_grams` , `weight_source` , `manual_raw_cost/manual_weight_grams/manual_cost_per_kg` , `cost_overridden` (שומר על ידנית מפני דריסת סנכרון), `odoo_id` , `image_url` , `is_active` , `odoo_archived` .
boms	רשומת מתכון (אחת לכל item). `yield_kg` , `recipe_type` (base/final), `sale_uom` (kg/unit), `ce_code` , עלויות ייצור: `labor_cost/overhead_cost/packaging_cost` , **snapshots**: `cost_per_kg/total_cost/wholesale_price/retail_price` , `pricing_formula_id` (נוסחה מו `archived` (נבדל ממחיקה), `version` , מיתוג: `me/description/allergens/is_spicy/serving_suggestion/servings_count/total_weight` .
bom_lines	רכיבי מתכון. `ingredient_item_id` (חו"ג או תת-מתכון), `quantity_kg` , `line_uom` , `waste_pct` (– 100), `step_number` , `price_per_kg_snapshot` , `line_cost` . UNIQUE(bom_id, ingredient_item_id).
bom_steps	שלבי הכנה. `step_number` , `step_name` , `description` .
bom_snapshots	היסטוריית גרסאות מלאה (JSONB) לכל שמירה.
cost_history	יומן עלויות append-only (מקור: odoo_sync) למעקב מגמות.
pricing_formulas	**מבנה "גבוה":** שורה לכל tier (wholesale/retail), מקושרות ב- `formula_uid` . `multiplier` , `rmula_expr` חופשי), `is_default` (יחיד), `name` . עמודות legacy `scope/scope_ref_id/priority` קיימות אך **לא ב**
reference_code_categories	קידומות קוד ייחוס. `prefix` (3–5 אותיות גדולות), `description` , `is_active` .
test_recipes	ארגז-חול. `status` ∈ {draft, pending}, `draft` (JSONB), `review_note` .
users	משתמשים. `role` , `can_view_prices` (tri-state: null=ברירת-מחדל לפי תפקיד), `password_hash` , `id` `is_active` .
audit_logs	יומן ביקורת append-only (התחברויות, שינויי משתמש/נוסחה, סנכרון, חישובי כמות).
categories	קטגוריות מוצר מ-Odoo.

---

4. מסכים ומודולים (Frontend)

ניווט בסרגל צד, מסונן לפי תפקיד. `ManagerRoute` (manager בלבד), `AdminRoute` (admin|manager).

מסך	נתיב	הרשאה	תיאור
התחברות	`/login`	ציבורי	שם משתמש + קוד.
דשבורד	`/dashboard`	manager	ספירות (בסיס/סופי/מוצרים/משתמשים/טסט), סטטוס סנכרון אחרון.
ספר מתכונים	`/book` , `/book/:id`	customer+	צפייה/חיפוש/סינון (אלרגן, חריף, קטגוריה), כרטיס מתכון, מחשבון כמויות, הדפסת דף-הכנה.
מתכוני מטבח	`/kitchen`	manager	מרכז ניהול: טאבים בסיס/סופי (עם ספירות), תצוגת כרטיסים/רשימה, בחירה מרובה + פעולות גורפות (מחיקה/ארכוב/ייצוא/הדפסה), צפייה בארכיון.
צפייה במתכון (ניהול)	`/recipes/view/:id`	manager	תצוגה מלאה: תפריט גלגל-שיניים (עריכה/הדפסה/ייצוא/ארכוב/מחיקה), **סקאלר כמות** (מחשב מחדש את כל הרכיבים כולל תתי-מתכון), קישור לתת-מתכון, שלבי הכנה (כולל של תתי-מתכון).
בונה מתכונים	`/recipe/new` , `/recipe/:id`	manager	יצירה/עריכה (פירוט בסעיף 4.1).
מתכוני טסט	`/test-kitchen`	admin	ארגז-חול עם רכיבים אד-הוק (מסומנים אדום עד שמזוהים).
ממתינים לאישור	`/pending-recipes`	manager	אישור/החזרה (עם הערה) של מתכוני-טסט שהוגשו.
Where Used	`/where-used`	manager	מאיתור: אילו מתכונים משתמשים ברכיב/תת-מתכון (עומק, ישיר/עקיף).
מוצרים	`/products`	admin	קטלוג Odoo: עלות/kg חיה, עריכת override (עלות/משקל), סינון ארכיון.
הגדרות	`/settings`	manager	טאבים: נוסחאות תמחור, משתמשים, סנכרון Odoo, קודי ייחוס.
לוגים	`/logs`	manager	יומן ביקורת עם סינון + ייצוא CSV.

4.1 בונה המתכונים (RecipeBuilder)

מצב real/test. שדות: שם, **קוד ייחוס** (כולל בחירת קטגוריה → מספר פנוי אוטומטי), Yield(kg), סוג (base/final), **יחידת מכירה** (kg/unit, רק final), אסטרטגיית תמחור (נוסחה מוצמדת או ברירת-מחדל), רכיבים (חיפוש, kg/g/L/ml/unit: **UOM**, % פחת), עלויות ייצור, שלבי הכנה, כרטיס מיתוג (תמונה/תיאור/אלרגנים/חריפות). **תצוגה חיה** של עלות/kg, עלות כוללת, סיטונאי וקמעונאי — מחושבת מקומית דרך `useBomCost` תוך **הרצת הנוסחה עצמה** (כולל עיגול).

4.2 ניהול מצב (Frontend)

- **useRecipeStore** (Zustand + persist) — טיוטת הבונה: lines, yield, recipeType, saleUom, multipliers + formulas, steps, מיתוג. נשמר ב-localStorage לטיוטות חדשות.
- **React Query** — מצב שרת (cache, invalidation). מפתחות עיקריים: `['boms', type, archived]` , `['formulas']` , `['pricing']` .
- **useToastStore**, **useModalStore** (drill-down).

---

5. מנועי ליבה (Backend)

5.1 מנוע עלויות — `costingService.js`

חישוב רקורסיבי: `line_cost = quantity_kg/(1-waste%) × cost_per_kg` ; `total_cost = Σ line + labor + overhead + packaging` ; `cost_per_kg = total_cost / yield_kg` . תת-מתכון →

רקורסיה. **זיהוי תלות מעגלית** דרך Set אבות. `recalculateAll()` ממיין טופולוגית (Kahn) ומחשב בסדר בטוח.
`calculationService.js` משרת את מחשבון הכמויות (סקיילינג לכמות יעד + רשימת קניות מצרפית).

5.2 מנוע תמחור ונוסחאות — `formulaEval.js` , `pricingService.js`

- **בחירת נוסחה** (לפי עדיפות): נוסחה מוצמדת על ה-BOM → נוסחת ברירת-מחדל (`is_default`) → fallback קשיח (×2.5/×5).
- **ביטוי חופשי בטוח** (parser רקורסיבי, ללא eval): `( ) ÷ × − +` , המשתנה `cost` , ופונקציות: `roundup/rounddown/round` (לשלם/לעשרוני) ו- `roundupto/rounddownto/roundto(x, step)` (לכפולות, למשל 5/10). דוגמה: `roundupto((cost*1.5)*1.07, 5)` .
- `applyFormula(expr, multiplier, cost)` — מריץ את הביטוי על העלות (מדויק, כולל עיגול), עם נפילה ל- `cost × multiplier` . מופעל גם על עלות/kg וגם על העלות הכוללת (per-yield).
- **resolver אצוותי** ב- `GET /boms` — שולף ברירת-מחדל פעם אחת, נוסחאות מוצמדות בנפרד (deduped) — מונע N+1 (שיפר טעינה מ-~שניות ל-~0.2 שנייה).

5.3 חילוץ עלות ומשקל מהשם — `weightExtractor.js` , `costResolver.js`

- **עלות:** manual → Odoo standard_price → raw_cost.
- **משקל/מידה:** manual → Odoo volume_weight → **רג'קס מהשם**: משקל ("1kg","200g"), נפח ("1l"), **כמות יחידות** ("6 50","units" יחידות") — מחלק את העלות ומחלץ מחיר ליחידה.
- **מוצרי יחידה (uom=Unit):** רק **כמות בשם** מחלקת; משקל/נפח בשם (קיבולת אריזה) **מתעלמים**; בלי מספר → עלות = מחיר ליחידה אחת.

5.4 סנכרון Odoo — `odooSyncService.js`

XML-RPC, שליפת מוצרים דו-לשונית (en/he) + ארכיון, scoping לחברה (`ODOO_COMPANY_*`), upsert גורף (UNNEST), שמירה על `cost_overridden` , יומן `cost_history` , תזמון cron (`ODOO_SYNC_SCHEDULE`), טיפול ב-TLS לשרתי staging.

5.5 ייבוא/ייצוא אקסל — `recipeIO.js` , `recipeIOService.js`

שורה לכל רכיב; **aliases** רחבים לכותרות (אנגלית/עברית/שגיאות כתיב); תמונות (URL/מוטמע/base64 חוזר); **קוד BAS-#### אוטומטי** לבסיס בלי קוד; שמירה לכל מתכון בטרנזקציה נפרדת (כשל אחד לא חוסם); **דו"ח ייבוא** מלא (נוצר/עודכן/נכשל + סיבות) + ייצוא CSV.

5.6 קודי ייחוס — `referenceCodeCategories.js`

תבנית `PREFIX-####` . `usedNumbersForPrefix` אוסף מספרים בשימוש מ- `boms` + `test_recipes` + `items` (פעילים בלבד) ומחזיר את **המספר הפנוי הנמוך ביותר** (ממלא חורים).

---

6. ממשק API (תמצית לפי ראוטר)

`auth` · (login) `boms` · (CRUD, summary, snapshots, calculate, bulk-delete/archive) `items` · (search, affected-recipes, pricing,) `products` · (recalculate) `pricing` · (GET, PATCH override) `reference-codes` · (CRUD, נוסחאות, resolve, default) `recipe-io` · (next) `test-recipes` · (template, import, export) `sync` · (CRUD, submit, send-back, promote) `users` · (status, odoo,) `audit-logs` · (CRUD, me, audit) `categories` · (costs) · (list, action-types) .

---

7. מה כבר קיים במערכת (סיכום)

✅ אימות + 3 תפקידים + נראות מחירים בצד-שרת · ✅ בונה מתכונים מלא (בסיס/סופי, UOM, פחת, שלבי הכנה, מיתוג) · ✅ קינון תתי-מתכון + חישוב עלות רקורסיבי + זיהוי מעגליות · ✅ מנוע נוסחאות חופשי (מכפיל/מע"מ/עיגול/כפולות) + עורך ויזואלי · ✅ סנכרון Odoo (דו-לשוני, ארכיון, override) · ✅ חילוץ משקל/נפח/יחידה מהשם + מחיר-ליחידה · ✅ ייבוא/ייצוא אקסל + דו"ח · ✅ קודי ייחוס אוטומטיים · ✅ מתכוני-טסט + workflow אישור · ✅ Where Used · ✅ מחשבון כמויות + סקאלר · ✅ ארכוב/מחיקה + פעולות גורפות · ✅ דשבורד, לוגים, ניהול משתמשים · ✅ יחידת מכירה (kg/unit) · ✅ resolver אצוותי (ביצועים).

8. מפת דרכים — מה לתקן/להוסיף (מתועדף)

🔴 P0 — קריטי (נכונות מחירים/נתונים)

1. **מחירים/עלויות שמורים מתיישנים.** `boms.cost_per_kg/total_cost/wholesale/retail` נשמרים בעת שמירה. כששינוי **עלות חומר-גלם** (סנכרון/override) או **שינוי נוסחה** קורה — המתכונים שמשתמשים בהם **לא מתעדכנים אוטומטית**. *פתרון:* טריגר אוטומטי ל- `recalculateAll` אחרי סנכרון Odoo, + **כפתור "חשב מחדש הכול" נגיש ב-UI** (כיום קיים רק כ-endpoint), + **אינדיקציה ויזואלית** "מחיר לא מעודכן" על הכרטיס. *(הערה: רשימת המתכונים כבר מחשבת תמחור חי; הבעיה ב-snapshots השמורים ובמסכי הצפייה/הדפסה.)*
2. **אזהרה לפני מחיקת/ארכוב רכיב בשימוש.** קיים `GET /items/:id/affected-recipes`, אך אין אזהרה בממשק לפני מחיקה/ארכוב של רכיב/תת-מתכון שמשמש מתכונים. *פתרון:* לפני מחיקה — להציג כמה מתכונים יושפעו ולחסום/לאשר.
3. **מניעת תלות מעגלית בזמן שמירה.** כיום נתפסת רק בזמן חישוב (אחרי שנוצרה) → כל חישוב עלות נכשל. *פתרון:* בדיקה ב- `saveRecipeBom` עם הודעה ברורה.

🟠 P1 — חשוב (שלמות workflow ו-UX)

1. **חשיפת "חשב מחדש הכול" + טריגר אוטומטי** (משלים את P0-1; כפתור בהגדרות→סנכרון).
2. **לולאת משוב מלאה למתכוני-טסט** — מצב "ממתין" עם הערות הלוך-ושוב ברור יותר.
3. **חיפוש טקסט חופשי ברשימות מתכונים בצד-שרת** (כיום סינון לפי סוג/ארכיון בלבד; חיפוש מקומי בלבד).
4. **דיווח שגיאות סנכרון Odoo ב-UI** — כיום כשלים חלקיים נכתבים ל-console בלבד.
5. **אינדיקציית "מחיר עודכן לאחרונה / לא מעודכן"** על כרטיס המתכון (משלים P0-1).

🟡 P2 — שיפורים (נוחות וניקיון)

1. **שכפול נוסחה (Clone)** במקום הקלדה מחדש.
2. **גרף/מסך היסטוריית עלויות** — הנתונים קיימים ב- `cost_history`, אין UI.
3. **ניקוי legacy** ב- `pricing_formulas` (עמודות `scope/scope_ref_id/priority` לא בשימוש).
4. **גודל קבצי ייצוא** — תמונות base64 מנפחות את ה-xlsx; לשקול קישורים/דחיסה.
5. **תיעוד/בהירות per-unit** — `sale_uom` הוא דגל תצוגה בלבד; לוודא עקביות בכל המסכים וההדפסות.
6. **בדיקות אוטומטיות** למסלולים קריטיים (תמחור, עלויות, ייבוא, round-trip תמונות) — כיום אין.

---

9. נספח — הרצה וסביבה

- **Backend:** `npm start` (ללא reload אוטומטי — **חובה restart אחרי שינוי backend/.env**). בריאות: `GET /api/health`.
- **Frontend:** Vite dev server (HMR), `npx tsc --noEmit` לבדיקת טיפוסים.
- **DB:** מרוחק (~90ms latency/שאילתה) — לכן חשוב להימנע מ-N+1.
- **משתני סביבה (.env, לא ב-git):** `DB_*`, `ODOO_URL/DB/USER/PASSWORD`, `ODOO_COMPANY_*`, `JWT_SECRET`, `ALLOW_DEV_LOGIN` + `DEV_ADMIN_*`, `ODOO_SYNC_SCHEDULE`.