Awesome—here’s a **detailed, step-by-step plan** you can follow **one 2–3h session at a time**. It’s built around your rules: **perimeter (no islands), global upgrades per type, persistent kitchen state, shop between runs, bots at L4, space expansions later**. Every step has a **clear “exit test”** so you know when to stop and move on.

**Phase A — Vertical Slice (Sessions 1–12)**

**Session 1 — Project prep & Base Scene**

**Do**

* Make a BaseKitchen scene with lighting, camera, player spawn, UI canvas.
* Create empty GameObjects: GameManager, UIManager, EconomyManager.  
  **Exit test:** Scene runs, player can move; no errors in console.

**Session 2 — Placement Slots (perimeter only)**

**Do**

* Add WallSlot components (Transform + slotId, sizeTag: 1x1/2x1, areaId).
* Place 8–12 wall slots around the room; no center slots.  
  **Exit test:** A gizmo/label shows each slot; you can FindObjectsOfType<WallSlot>() and list them.

**Session 3 — Item & Upgrade data (SO)**

**Do**

* ScriptableObjects: ItemDef (type, sizeTag, prefab per tier), UpgradeTier (speed/capacity numbers), RecipeDef, LevelConfig.
* Create 3 core items: **CuttingCounter**, **PlateCounter**, **Toaster** (or Grill/Fryer—pick your L1).  
  **Exit test:** You can select an ItemDef and see its tier prefabs & stats.

**Session 4 — Upgradable prefabs (global)**

**Do**

* One prefab per item type with UpgradeController (switch mesh/stats by tier), or separate prefabs per tier.
* Global upgrades dictionary: Dictionary<ItemType,int> globalUpgrades.  
  **Exit test:** Calling ApplyUpgrade(Cutting, 2) updates **all** cutting counters in the scene.

**Session 5 — Save/Load (persistent kitchen)**

**Do**

* Data model:

text

CopyEdit

KitchenSave {

coins

ownedItems: [ { slotId, itemType } ]

globalUpgrades: { itemType -> level }

unlockedLevel

}

* JSON save to Application.persistentDataPath.  
  **Exit test:** Place two owned items in code, save → restart → they respawn in the same slots with upgrade visuals.

**Session 6 — Replace-in-slot flow (buy new without space)**

**Do**

* Temporary dev UI: list “Buy New Item” → opens **slot picker** of occupied slots → replace selected slot’s item.  
  **Exit test:** Buy CuttingCounter → choose a slot → item swaps; save persists.

**Session 7 — Orders & Timers v1**

**Do**

* OrderQueue: spawns orders from RecipeDef with a countdown.
* On deliver before timeout → add coins; on expire → fail count.  
  **Exit test:** Orders appear with timers and expire correctly.

**Session 8 — Cooking interactions v1**

**Do**

* Minimal flow for 2 simple recipes (e.g., Tomato Salad + Toasted Sandwich).
* Interactables: pick up, drop, use counter/appliance → result item.  
  **Exit test:** You can complete both recipes end-to-end manually.

**Session 9 — Delivery & Scoring**

**Do**

* Delivery counter validates plate contents vs order.
* Rewards: base coins + speed bonus; basic streak bonus (optional).  
  **Exit test:** Delivering gives coins; UI shows your coin total rising.

**Session 10 — Shop v1 (Upgrades)**

**Do**

* Shop UI with tabs: **Upgrades** (global), **Buy New** (replace slot).
* Prices from ItemDef; coin check; confirm → apply & save.  
  **Exit test:** Upgrade Cutting to Tier 2; all cuttings visually/stat-update instantly; coins reduce.

**Session 11 — Level 1 layout & loop**

**Do**

* Build L1 with placeholders: containers + Cutting + Toaster/Grill + Plates + Delivery.
* Connect “End Run → Shop → Start Run” flow.  
  **Exit test:** You can play L1, earn coins, shop, replay—no manual scene hacking.

**Session 12 — Clean pass & guardrails**

**Do**

* Validate aisle width ≥ **1.2m** (simple ray/box checks).
* Filter slots by sizeTag when buying.
* Add simple pause/restart.  
  **Exit test:** You can’t place a 2×1 item in 1×1 slot; aisle guard prevents blockages.

**Phase B — Scale to L2–L3 (Sessions 13–18)**

**Session 13 — New recipes & counters for L2**

**Do**

* Add Blender or second heat method per your plan.
* Add 1–2 L2 recipes; create upgrade tiers for any **new** counter types.  
  **Exit test:** L2 plays; new counters appear in shop; upgrades affect stats.

**Session 14 — Balance pass 1**

**Do**

* Tune: order spawn rate, timers, coin rewards, upgrade prices (target upgrade payback **3–5 runs**).  
  **Exit test:** A fresh player can reliably afford the first upgrade within 3–5 runs.

**Session 15 — L3 content**

**Do**

* Add Grill/Fryer (if not in L1/L2) and corresponding recipes.
* Ensure global upgrades exist for these types before shipping L3.  
  **Exit test:** L3 plays; upgrades feel meaningful.

**Session 16 — UX passes**

**Do**

* Ticket UI shows time remaining.
* “Ready” highlight on finished cooked items; error toasts (e.g., wrong plate).  
  **Exit test:** No confusion on what to do next during a run.

**Session 17 — Economy telemetry**

**Do**

* Log orders/min, fails, average time-to-finish; save a small CSV/JSON for dev.  
  **Exit test:** After a run, you can read a file with basic metrics.

**Session 18 — Bugfix buffer**

**Do**

* Fix top 3 friction points you’ve noted.  
  **Exit test:** Clean console; stable 30–60 fps on your target device.

**Phase C — Helpers (unlock at L4) (Sessions 19–24)**

**Session 19 — Helpers framework**

**Do**

* BotDef SO (tiers, costs) + BotState in save.
* Shop → **Helpers** tab; purchase places a **base station in a WallSlot**.  
  **Exit test:** Buy Plate Bot; base station appears; persists.

**Session 20 — Plate Bot v1**

**Do**

* Keeps N plates stocked on designated plate counters (buffer = 3).
* Upgrades: speed, buffer.  
  **Exit test:** With bot tier 1, plate counters rarely run empty.

**Session 21 — Delivery Bot v1**

**Do**

* Moves complete, plated orders from “Ready” to Delivery counter.
* Upgrade: carry 2; smarter priority.  
  **Exit test:** Orders arrive faster; less player walking.

**Session 22 — Cook Bot v1 (tight scope)**

**Do**

* Services 1 assigned appliance + input/output counters.
* Doesn’t fetch ingredients; only runs appliance & ejects output.  
  **Exit test:** If input exists, appliance never idles; bot places output reliably.

**Session 23 — Balance pass 2 (with bots)**

**Do**

* Re-tune coin rewards/prices so bot payback stays ~3–5 runs.  
  **Exit test:** Owning a bot increases orders/min ~15–25% at same skill.

**Session 24 — L4 content**

**Do**

* Add L4 recipes/counters; show a **one-run bot rental** tutorial.  
  **Exit test:** Rental converts into purchase for some players (in your testing).

**Phase D — L5–L8 Mechanics (Sessions 25–36)**

*(Repeat the same 4-session cycle per level: A) New counters/tiers → B) Layout/wiring → C) Balance → D) Bugfix)*

**Example for L5 (Sessions 25–28)**

* **25:** Add Oven + Dough Mixer + tiers (faster bake, double rack, etc.)
* **26:** Build L5 layout; wire recipes.
* **27:** Balance (timers ↑, rewards ↑, prices ↑; keep payback constant).
* **28:** Bugfix & polish.

**L6 (29–32), L7 (33–36)**

* L6: Sous Vide + Wine Pour (tiers defined before release).
* L7: Multi-meat grill combos; ensure upgrades address throughput.

**L8 (37–40)**

* Add Soup Pot; late-game difficulty + coin scaling; final tuning.

**Exit test after each level:**

* Featured counters have **tiers** & show in shop.
* Level beatable with reasonable upgrades; economy doesn’t stall.

**Phase E — Space Expansions (Optional, after L4) (Sessions 41–46)**

**Session 41 — Areas & slots**

**Do**

* Define ExpansionArea (SO: id, price, slots it unlocks, pre-baked NavMesh).  
  **Exit test:** Buying an area toggles new wall slots alive.

**Session 42 — NavMesh swap**

**Do**

* Enable pre-baked NavMeshData per area on unlock.  
  **Exit test:** Bots/players path into new room; no baking at runtime needed.

**Session 43 — Shop: Space tab**

**Do**

* “Unlock Space” card in shop; coin/ad alternatives.  
  **Exit test:** New purchases can go into **empty wall slots** (no replacement).

**Sessions 44–46 — Content & balance**

**Do**

* Add a couple of expansion-only slot bonuses (e.g., 2×1 appliances).
* Adjust economy for bigger kitchens.

**Always-On Guardrails**

* **Placeholders first.** Don’t wait for art—use gray boxes.
* **Global upgrades only** for functional counters (your rule).
* **One change per level.** Introduce a mechanic on Lx; save extras for Lx+1.
* **Aisle width check ≥ 1.2 m** before allowing placement.
* **Slot-size filter** when buying (prevents invalid placements).
* **Payback window:** Upgrades/bots recoup in ~3–5 runs.

**Tiny implementation cheatsheet (copy/paste)**

**ScriptableObjects you’ll need**

* ItemDef (type, sizeTag, tierPrefabs[], tierStats[])
* RecipeDef (inputs, steps, output, time)
* LevelConfig (recipes, order rate, length, rewards)
* ExpansionArea (price, slotIds, navmesh asset)
* BotDef (tiers, costs, capabilities)

**Save structs**

csharp

CopyEdit

class KitchenSave {

public int coins;

public int unlockedLevel;

public Dictionary<string,int> globalUpgrades;

public List<OwnedItem> ownedItems; // {slotId,itemType}

public List<BotState> bots; // {botId,tier,stationSlotId}

public int unlockedAreasMask;

}

**Job system (bots)**

* Queue types: PlateTask, DeliveryTask, CookTask.
* Priorities: expiring plate > idle appliance > plate buffer.

**How to use this plan day-to-day**

* Work **top to bottom**: don’t jump phases.
* Stop when you hit the **exit test**—that’s your “done”.
* If you get blocked >30 min, **stub it** (fake data/placeholder) and move on.
* Keep a **running “Later” list**; only pull from it on bugfix sessions.

If you want, I can convert these sessions into a **printable checklist** (same steps, with tick boxes and mini exit tests) you can paste into Word.