

Tenancy Receipt Book

Capstone 28 · PropTech · Talodabi Academy

1 · Project overview

Landlord-tenant rent receipt trust. Not property marketplace.

Deliverable	Full mobile MVP in Figma (not code)
Platform	Mobile app · Android-first · portrait
Screens	11 screens + 1 unhappy path (12 frames)
Features	3 core features
Bottom navigation	Properties · Receipts · Tenants · Profile
Prototype	4 linked flows

Build order: Auth (screens 1–4) → nav tabs (5–8) → core flow (9–11) → unhappy path → link prototypes.

2 · Problem & primary user

Rent is often cash with disputed records. UK has tenancy deposit schemes; Nigerian tenants struggle to prove payment.

Primary user

"Mr. Adebayo, a landlord in Abeokuta with six tenants who writes receipts in a notebook tenants lose."

Geographic scope: National-level

3 · Goals

User goal: Issue digital receipts tenants can save and forward.

Business goal: Landlords pay monthly; property managers buy multi-unit plans.

Success: Landlord issues receipt; tenant views history.

4 · Market context

QUESTION	ANSWER
Who pays	Landlords pay monthly; property managers buy multi-unit plans.
Why not WhatsApp	Handwritten receipts get lost; WhatsApp photos are disputed.
Design focus	Landlord issues PDF receipt; tenant saves history on phone.

Competitors to check:

- **Paper receipt book** (default)
- **Bank transfer screenshot** (no formal receipt)
- **Property proptech** (often Lagos-focused)

5 • Validate before you design

Interview 3–5 people who match your primary user. Update this brief if interviews contradict it. Do not start hi-fi until you complete the confirmation below.

FIELD	YOUR ANSWER
Validated user (name, age, city)	
Problem in one sentence	
One interview quote	
Success metric for your test	
Primary colour and tone	
One competitor you checked	

Interview prompts

- "Walk me through the last time this happened."
- "What did you use instead? WhatsApp, paper, or nothing?"
- "What would make you trust this on a cheap Android phone?"
- "What would stop you from using this every week?"

6 • Features & scope

v1 • three features

1. Add tenant and property unit
2. Issue receipt after each rent payment
3. Tenant views and downloads receipt history

Out of scope

- Eviction workflow
- Mortgages
- 100-unit enterprise

7 · Screens

Design every row in wireframe, high-fidelity UI, and prototype (where the flow applies).

#	LAYER	SCREEN	WHAT TO SHOW
1	Auth	Splash Screen	App logo, name, and tagline for Tenancy Receipt Book.
2	Auth	Onboarding (1 screen)	Explain the app in one sentence. One Get Started button.
3	Auth	Sign Up Screen	Name, phone number, state, and PIN. No email required.
4	Auth	Sign In Screen	Phone number and PIN. Forgot PIN link.
5	App shell	Properties (tab)	Property list with rent due dates and paid/unpaid badges. Show bottom nav with Properties active.
6	App shell	Receipts (tab)	All receipts this month. Issue Receipt button. Show bottom nav with Receipts active.
7	App shell	Tenants (tab)	Tenant cards with unit, phone, and receipt history link. Show bottom nav with Tenants active.
8	App shell	Profile (tab)	Landlord name, phone, and number of units managed. Show bottom nav with Profile active.
9	Core flow	Issue receipt · PDF	Rent amount, month, payment method. Show bottom nav on post-sign-in screens.
10	Core flow	Receipt preview	PDF preview before share. Show bottom nav on post-sign-in screens.
11	Core flow	Tenant · Receipt shared	Tenant can view receipt history. Show bottom nav on post-sign-in screens.

Layers: Auth = 1–4 · App shell = 5–8 · Core flow = 9–11

8 · Prototype flows

Link these in Figma. Examiners should click through without your voice-over.

Flow 1 · App entry

Splash → Onboarding → Sign Up or Sign In → first main tab

Flow 2 · Core task (all three features)

Sign In → Properties tab → Receipts tab → Issue receipt → Tenants tab

Core flow screens: Issue receipt · PDF · Receipt preview · Tenant · Receipt shared

Flow 3 · Navigation

Link every nav tab so examiners can tap each bottom nav item. Tabs: Properties · Receipts · Tenants · Profile

Flow 4 · Unhappy path

Trigger the unhappy path edge case and link to that screen.

9 · Unhappy path

Design this as its own screen state (frame 12):

Landlord issues receipt for month already paid. Show: This month already has a receipt. View or edit?

Link in **Flow 4**.

10 · Design system

RULE	DETAIL
Colour	You choose. Keep it simple and readable.
Type	You choose. Keep body text easy to read on a phone.
Tone	Design for your user, not for a tech audience.

Reuse on every screen:

- Primary button (active and disabled states)
- Input field (empty, filled, and error states)
- Card or list item used across the app
- Navigation bar with correct labels and icons
- Empty state: what the screen looks like with no data

11 · Submit & defense

Deliverables

#	DELIVERABLE	WHAT GOOD LOOKS LIKE
1	Wireframes	Every screen in this brief, plus the unhappy path. Layout and labels clear.
2	High-fidelity UI	Same frames polished. You choose colour and type. Bottom nav on every post-sign-in screen.
3	Prototype	Four linked flows (entry, core task, navigation, unhappy path). Examiners can use the app without your voice-over.
4	AI prototype	Figma Make: at least 2 screens. Share link.
5	Case study (1 page)	Problem · User · Who pays · Competitors · Solution · One design decision
6	Presentation (6–8 slides)	Name · Problem · User · Solution · Key screens · Flow demo · One decision

Before you present

- ☐ Every screen designed in wireframe and hi-fi
- ☐ Unhappy path screen designed and linked in the prototype
- ☐ Four prototype flows linked end to end
- ☐ At least one empty state on a list tab
- ☐ Shared components reused: button, input, card or row, bottom nav
- ☐ Case study names who pays and one competitor you checked
- ☐ You can explain one design decision in critique (why, not what)
- ☐ Completed 5 user interviews or 1 usability test
- ☐ Checked at least one competitor listed under Market context

© Talodabi Academy · For enrolled students · Not for redistribution

