

# Borehole Monitor

Capstone 40 · PropTech · Talodabi Academy

## 1 · Project overview

Compound pump schedule and log.

|                          |                                         |
|--------------------------|-----------------------------------------|
|                          |                                         |
| <b>Deliverable</b>       | Full mobile MVP in Figma (not code)     |
| <b>Platform</b>          | Mobile app · Android-first · portrait   |
| <b>Screens</b>           | 11 screens + 1 unhappy path (12 frames) |
| <b>Features</b>          | 3 core features                         |
| <b>Bottom navigation</b> | Pump · Schedule · Alerts · Settings     |
| <b>Prototype</b>         | 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

Shared boreholes serve estates nationwide. UK has municipal water; Nigerian estates manage pumps manually.

### Primary user

"Alhaji Musa, a 55-year-old estate chairman in Minna managing pump hours for 12 households."

**Geographic scope:** National-level

## 3 · Goals

**User goal:** Log pump runs and see fair usage schedule for residents.

**Business goal:** Estate managers pay subscription; IoT sensor upsell partners.

**Success:** User logs pump and sees schedule conflict.

## 4 · Market context

| QUESTION                | ANSWER                                           |
|-------------------------|--------------------------------------------------|
| <b>Who pays</b>         | Estate SaaS plus IoT sensor upsell partners.     |
| <b>Why not WhatsApp</b> | Chairman tracks pump hours in a notebook.        |
| <b>Design focus</b>     | Pump log plus fair schedule plus overtime alert. |

### Competitors to check:

- **Manual pump roster** (default)
- **WhatsApp complaints** ( unstructured)
- **Smart water IoT** (enterprise pricing)

## 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. Pump on/off log with duration
2. Weekly schedule agreed by estate
3. Alert when pump runs outside schedule

## Out of scope

- Municipal water billing
- Full IoT hardware design
- 100-estate 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 Borehole Monitor.                                     |
| 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 | Pump (tab)            | Today pump hours and Log Pump start/stop button. Show bottom nav with Pump active.    |
| 6  | App shell | Schedule (tab)        | Calendar of allowed pump windows per block. Show bottom nav with Schedule active.     |
| 7  | App shell | Alerts (tab)          | Overtime pump alerts and chairman notify toggle. Show bottom nav with Alerts active.  |
| 8  | App shell | Settings (tab)        | Estate name, block count, and chairman contact. Show bottom nav with Settings active. |
| 9  | Core flow | Pump · Session logged | Start/stop or duration saved. Show bottom nav on post-sign-in screens.                |
| 10 | Core flow | Schedule · Block view | Allowed pump window for one block. Show bottom nav on post-sign-in screens.           |
| 11 | Core flow | Alert · Overtime      | Pump ran outside schedule warning. 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 → Pump tab → Session logged → Schedule tab → Alerts tab

Core flow screens: Pump · Session logged · Schedule · Block view · Alert · Overtime

## Flow 3 · Navigation

Link every nav tab so examiners can tap each bottom nav item. Tabs: Pump · Schedule · Alerts · Settings

## 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):

*Two residents log conflicting pump times. Show: Conflict logged. Chairman review needed.*

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

## 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

