RAD-SMART: Phase 2 Final Submission

Health-a-thon 2026 · Cancer track · Use case 04: Clinic Operations & Patient Flow · 27 October 2026

> **Deadline: Sunday 8 November 2026.** Top 30 announced by 14 November; finale on 28 November at IIT Bombay. The eight sections below follow the Phase 2 template. Each has a **full version** and a **short version** in case the form limits length. Replace every [square bracket] before submitting: they mark evidence only the department can give (the technologist trial and the native-speaker ratings).

> **Everything described as built in this document runs today** in the CARE plugin in `app/`, is shown in the demo video, and is **live on Microsoft Azure** for anyone to try: https://radsmart.braveglacier-5f9807f7.uaenorth.azurecontainerapps.io/try/ Section 4.3 lists what is not built yet. Every patient in the app, the video and the evaluation is synthetic; the department's records are used only as anonymised totals.

---

1. Team Overview & Problem Statement

Team

Member	Role
Dr Akshay Dinesan, Manipal	Team leader and doctor partner
Abhinand T M	Technical lead
P S Raghotham Rao	Lead engineer: scheduling engine, CARE plugin, AI layer
Dr Shirley Lewis Salins	Doctor partner
Dr Umesh Velu	Doctor partner

The clinicians work in the department RAD-SMART is built for, so the rules in the software are the department's own: the technologists and doctors answered our questions in writing on 21 September and again on 24 October, and both sets of answers are now in the code.

Problem

The department has **one linear accelerator** (an Elekta Versa HD) treating **70–90 patients a day**, from 08:30 until about 01:00. A radiotherapy course means coming every weekday for one to seven weeks, usually with a family member. Appointment times are given by hand, without knowing how long each session takes, so too many patients are asked to come at once and the delay builds through the day.

The department's own records (55 working days, October–December 2024, 3,137 patient visits, used only as anonymised totals) show:

- **one patient in ten waits more than 109 minutes** from arrival to treatment;
- **22% are treated more than an hour after their appointment**, and only 29% within 15 minutes of it;
- a machine breakdown is handled by phone calls and guesswork, and patients already on a bus are not told.

69% of India's radiotherapy facilities have only one machine, so this is the common case, not ours alone. A second machine arrives here in 2027; until then, and in most of India for longer, the only way to cut waiting is better use of the machine already there.

> **Short version.** One radiotherapy machine treats 70–90 patients a day. Times are given by hand without knowing session lengths, so one patient in ten waits over 109 minutes and 22% are treated over an hour late, every day of a 1–7 week course. Breakdowns are handled by phone. 69% of Indian radiotherapy centres have

> one machine. Team: Dr Akshay Dinesan (team leader, doctor partner), Abhinand T M (technical lead), P S Raghotham Rao (lead engineer), Dr Shirley Lewis Salins and Dr Umesh Velu (doctor partners).

2. Target Users & Solution

Users

User	What they do with RAD-SMART
Radiation oncologist	Enters a course from the patient's visit in CARE, with the patient's preferred part of the day; approves its first day and one standing daily time; can ask for a plain-language explanation in the patient's language
Radiation therapy technologist	Runs the day board (arrived, in room, out) with a live queue, taking a ready patient in out of turn when another is late; declares breakdowns and approves the re-plan; sends patient messages on WhatsApp; agrees better times with patients
Reception	Checks patients in by scanning the QR code on their phone
Patient and family	Open one private link on their phone: today's time, changes, messages in their language with a voice note, the check-in code, and four reply buttons, including a morning, afternoon or evening preference
Head of department	Sees on Insights whether long waits are falling

Solution

RAD-SMART is an assistive scheduler built **inside CARE**, the open-source hospital information system of the Open Healthcare Network, as a plugin with no change to CARE's own code. It works in four steps:

1. **Predict** how many minutes each session will take on the machine.
2. **Plan** the day with an optimiser that cannot break the department's hard rules (for example, new patients finish before 17:00; gap-sensitive courses are never dropped), giving each course one standing time for its whole course.
3. **Re-plan** in about 10–14 seconds when the machine breaks down, following the department's own downtime rules (up to 60 minutes: everyone is treated the same day; longer: those still at home are told first).
4. **Tell** patients in Kannada, Malayalam, Hindi, Konkani or English (Tulu speakers read Kannada script), using Sarvam AI, on a phone page and by WhatsApp.

A person approves every change. RAD-SMART decides only *when* a patient is treated, never *whether* or *how*.

> **Short version.** Users: radiation oncologists (enter and approve a course), technologists (run the day, breakdowns, messages), reception (QR check-in), patients and families (a phone page in their language), and the head of department (Insights). RAD-SMART is a plugin inside CARE. It predicts session minutes, plans the day with an optimiser that cannot break the department's rules, gives each course one standing time, re-plans in seconds after a breakdown, and tells patients in their language with Sarvam AI. A person approves every change.

3. Working MVP & AI Approach

3.1 What runs today

The MVP is a working application, not a mock-up: a Django backend plugin and a React frontend plugin loaded into CARE (about 3,600 lines of Python and 3,800 lines of TypeScript), on top of the scheduling engine (about 3,700 lines of Python). It is packaged as a Docker container and **deployed live on Microsoft Azure** (Section 7), so the judges can use the same build the department would pilot.

Page	What it does
RADSMART button on the patient's visit	Opens the patient's courses; shows the daily time once approved
New course	The prescription and a priority checklist (hard constraints and soft goals, including the preferred part of the day); *Save and propose slots* returns a first day and a standing time, with reasons and alternatives
Course	Approve, explain in plain words (English and the patient's language), the patient's link and QR code, the preferred part of the day, requests and offers of another time, the messages sent to this patient
Today	The machine card with a live queue (in the room, *Call next*, on time or running behind), the live rule check, the timeline, the patient list with arrival / in room / out, late and out-of-turn badges, RAD-SMART's better-time offers, and patients' replies
Breakdown	Declare a fault and its estimate; RAD-SMART proposes what happens to each remaining patient under the department's rules; a person approves; messages are drafted automatically
Messages	Every message drafted after an approval, translated, with a voice note; *Open WhatsApp* opens the chat with the text filled in; *Mark sent*. In the online demonstration a message can also be sent to the visitor's own Telegram, to show it arriving on a phone
Check-in	Reception scans the patient's QR code (USB scanner or camera) or types the six-letter code
Insights	Waiting times and punctuality from the times the board records
Patient page (no app, no login)	Today's time, any change after a breakdown, the next five treatment days, messages in the patient's language with a voice note, the check-in QR code, and four buttons: *on my way*, *I will be late*, *I cannot come today*, *I want a different daily time* (morning, afternoon or evening)

3.2 The AI approach: each part does what it is best at

Part	Method	Why this method
Session-minute prediction	Gradient boosting (scikit-learn) on technique, site, fraction number, new start, accessories and patient needs	Learns how long a session takes; the day board's *In room* and *Out* taps now record the real minutes it will be retrained on
Planning and re-planning	Mixed-integer optimisation (SciPy with HiGHS), with the department's rules as hard constraints and preferences as weighted soft goals	Gives a **guarantee** that rules hold, which a learning model or a chatbot cannot; an independent checker re-verifies every plan
Digital twin	A simulation calibrated on the department's records (it reproduces today's long wait: 106 against 109 minutes)	Tests a plan, and measures with and without RAD-SMART, before any patient is affected
Translation	Sarvam Translate (`sarvam-translate:v1`)	Indian languages, including Konkani
Voice notes	Sarvam Bulbul v3 text to speech	Many patients and carers read little; a voice note in their language can be played on any phone
Plain-language explanation	Sarvam-105B, given only the engine's own facts and reasons	Explains *why this time* to the doctor and patient in plain words

We chose this hybrid after scoring five approaches (a fixed timetable, a chatbot alone, optimisation alone, reinforcement learning and the hybrid): only the hybrid can both guarantee the rules and talk to patients in their language. **The language models never make a scheduling decision.** They only translate and explain decisions the optimiser has made and a person has approved, and every output is checked (Section 6).

> **Short version.** A working CARE plugin (Django + React) on our scheduling engine: a RADSMART button on the visit page, New course, Course, Today, Breakdown, Messages, Check-in and Insights pages for staff, and a no-login phone page for patients. AI: gradient boosting predicts session minutes; a mixed-integer optimiser plans and re-plans with the rules as hard constraints, checked independently; a calibrated digital twin tests plans; Sarvam Translate, Bulbul v3 and Sarvam-105B translate, voice and explain approved times in Kannada,

> Malayalam, Hindi and Konkani. Language models never make scheduling decisions, and every output is checked.

4. Build Sprint Progress

4.1 Milestones

Date (2026)	Milestone
19 Sep	Round 1 entry: engine, digital twin, results page, deck
21–22 Sep	The department's written answers built in; the twin calibrated on their 55 days of records
5 Oct	Shortlisted; the build sprint starts
21 Oct	Technical guide and mathematics notes
24 Oct	The department's second set of answers (roles, standing slots, breakdown rules, the screens they need) built into the engine and **into a working application inside CARE**; with/without impact analysis
25 Oct	User guide; demo video, narrated with Sarvam Bulbul; **Sarvam AI layer, patient messages, patient phone page, QR check-in and plain-language explanations built**; Sarvam evaluation; automated safety tests
26 Oct	**Deployed live on Microsoft Azure** in a Docker container (CARE's production build, managed PostgreSQL, HTTPS), with one-click demonstration roles and messages to Telegram; demo video re-recorded on the live server
27 Oct	From clinical feedback, **the preferred part of the day** with better-time offers, and **a live queue for early and late patients**; 45 automated tests, the engine's run by GitHub on every change
[by 7 Nov]	[Technologist trial on the demo; native-speaker rating of translations]

4.2 What changed because of the department's feedback

- **Standing slots.** The department asked for one fixed daily time per course instead of a new time every day. Every course now gets one time that is free on *every* treatment date and keeps every rule.
- **New-start rule.** New patients start and finish before 17:00, and 150 minutes before 17:00 is kept free for future new starts.
- **Breakdown rules.** Up to 60 minutes: everyone is treated the same day. Over 60 minutes: patients still at home are told not to travel, those already waiting are treated first, and missed fractions are added at the end of the course. The machine starting late moves new starts to the next working day.
- **Roles.** Only a doctor can enter or approve a course; a technologist runs the day and approves day changes.
- **Languages and WhatsApp.** The department confirmed that patients use WhatsApp, and that most speak Kannada or Tulu. Messages therefore open in WhatsApp in the patient's language; Tulu speakers get Kannada, which they read.
- **Preferred part of the day, and patients out of turn (27 Oct).** From clinical feedback: a patient's morning, afternoon or evening now replaces "older patients earlier", and as courses end RAD-SMART offers staff a slot inside it, which they agree with the patient. The day board keeps the plan as an order: it shows whom to call next among those here, and a technologist can take a ready patient in out of turn, with the reason recorded, never blocked.
- **From testing**, we fixed four problems before anyone saw them: translations in Indian scripts attach suffixes to times (for example ೆ್ in Kannada), which our first time check missed; Konkani translated "the next working day" as "next week", so messages now name the actual date; the explanation used the proposed time instead of the time in force; and one patient button was ambiguous in Kannada, so its English wording was simplified.

4.3 Built, and not yet built

Feature	Status
Course intake with standing slots, approval by a doctor	**Built**
Day board with arrival, in room, out, live rule check	**Built**
Breakdown re-planning under the department's rules, with approval	**Built**
Requests for another time, decided by staff	**Built**
Preferred part of the day, with better-time offers as slots free up, agreed by staff with the patient	**Built**
Live queue for early and late patients: call next, running behind, out of turn with a reason	**Built**
Insights from recorded times	**Built**
Messages in Kannada, Malayalam, Hindi and Konkani (Tulu speakers get Kannada), with voice notes, opened in WhatsApp	**Built**; a technologist presses send
Patient phone page with replies and QR check-in	**Built**
Plain-language explanation with a time check	**Built**
Production-grade deployment: one Docker image (CARE's production build, nginx, gunicorn), live on Microsoft Azure with managed PostgreSQL and HTTPS, deployed by one script	**Built and running**
Hands-on online demonstration: one-click roles, a fresh synthetic day on demand, messages to Telegram	**Built and running**
Sending and receiving automatically through the WhatsApp Business API	Not yet: needs the hospital's Meta business verification and approved templates
Spoken replies from patients (speech to text)	Not yet
A Tulu voice	Not possible with Sarvam today; staff can record one. Konkani has text but no Sarvam voice
Importing today's Excel lists and paper prescriptions	Not yet; courses are entered on the New course form
Changing rules in plain language	Not yet; rules are plugin settings changed by the developer
Reading the oncology information system (MOSAIQ / ARIA)	Not yet; planned read-only
Retraining the session-minute model on real minutes	Waits for pilot data; the board records them from day one
Two machines	Capacity forecast only; live planning is for one machine

> **Short version.** Since Round 1 (19 Sep) we built the department's two sets of answers into the engine (standing slots, the 17:00 new-start rule, the 60-minute breakdown rule, roles) and, on 24–25 Oct, a working CARE application with Sarvam messages in four Indian languages (Tulu speakers get Kannada), a patient phone page, QR check-in, plain-language explanations and automated safety tests. On 26 Oct it went live on Microsoft Azure in Docker, open for anyone to try. Testing found and fixed four translation and wording problems. Not yet built: automatic WhatsApp sending, spoken replies, a Tulu voice, Excel and prescription import, plain-language rule editing, OIS integration.

5. Evaluation & User Validation

5.1 Scheduling results

Two datasets, kept apart.

The department's own 55 days, re-planned in the twin calibrated on them:

	Without (recorded)	With RAD-SMART
90th-percentile wait	108.6 min	**70.4 min (−35%)**
Treated more than 1 hour late	22.2%	**5.7%**
Treated within 15 min of the appointment	28.9%	**49.5%**
Busiest days (70+ patients), 90th-percentile wait	130 min	**70 min (−46%)**

Synthetic patients, same patients and events with and without:

Scenario	90th-percentile wait, without → with
12 weeks with standing slots (about 63 patients a day)	82 → **49 min (−40%)**
One busy day, 87 patients	210 → **76 min**
150-minute breakdown at 10:00: plan kept vs live re-plan (300 days)	210 → **110 min**; median 104 → **39 min**; new plan in **14 s**

An independent checker found **no broken hard rule** on any evaluation day.

5.2 AI component results

Component	Test	Result
Session-minute prediction	2,000 held-out synthetic sessions	Mean absolute error **2.15 min**, against 2.62 for a table of averages
Optimiser	87-patient day, about 4,800 variables	**Proven optimum in 7.1 s** on a laptop
Sarvam Translate	7 message templates × 4 sets of times in each of Kannada, Malayalam, Hindi and Konkani (Tulu speakers get the Kannada text), each translated back to English	**112 of 112 kept every time** (100%); back-translation chrF median 76–80 (lowest 67); median 1.3–2.0 s per message
Sarvam-105B explanations	20 synthetic plans	**20 of 20 accepted**, **0 invented times**, 16 state both come-at times; median 1.5 s per answer
Safety guards	37 automated tests on the plugin (no Sarvam calls), plus 8 engine tests run by GitHub on every change	**All pass**: times read in every script, changed times rejected, invented times rejected, English fallback, rate limits retried, no phone number for demo patients

Full tables, the lowest-scoring messages and a sample explanation: `docs/evaluation/sarvam_evaluation.md` (regenerated by `tools/evaluate_sarvam.py`).

5.3 User feedback and testing

- **Department feedback on the design** (21 Sep, 24 Oct): the technologists' and doctors' written answers defined the roles, standing slots, breakdown rules and screens; Section 4.2 lists what changed.
- **Technologist trial**: [N technologists ran the demo day on DATE: recording arrivals, a breakdown, sending messages. What they said: ... What we changed: ...]
- **Native-speaker rating**: every translation and its back-translation is in `docs/evaluation/sarvam_translations_for_rating.csv` for speakers of each language to rate 1–5. [Results:

Kannada x.x, Tulu x.x, Malayalam x.x, Konkani x.x, Hindi x.x. Messages rated below 4 were reworded.]

- **Patient view**: [optional: N patients or carers looked at the phone page and voice note. What they said: ...]

5.4 Limitations

- **Replay, not a trial.** The 35% comes from re-planning recorded days in the twin, not from live use. The pilot measures it for real.
- **Synthetic session lengths.** The department's records have no session minutes, so the prediction model is trained on synthetic sessions until the board has recorded real ones.
- **Patient behaviour.** The gain assumes patients stop arriving very early once times are kept. If they keep today's habits, the busy-day 90th percentile is 110 minutes rather than 76, still below the busiest recorded days.
- **Costs of standing slots.** The evening runs about two hours later (still inside the department's hours), one new patient in five starts later by under half a day on average, and requested times are kept less often.
- **Translation.** Back-translation is a rough meaning check, not a substitute for native speakers. Sarvam has no Tulu model.

> **Short version.** On the department's 55 recorded days, re-planned in a calibrated twin, the 90th-percentile wait fell from 108.6 to 70.4 min (−35%) and over-an-hour lateness from 22.2% to 5.7%. Over 12 synthetic weeks: 82 → 49 min (−40%). After a 150-min breakdown, live re-planning cut the median wait from 104 to 39 min, with a new plan in 14 s. No hard rule was broken. Prediction error 2.15 min; optimum in 7.1 s. Sarvam translations kept every time in 112 of 112 messages across four languages; 20 of 20 explanations accepted with no invented time; 45 automated tests pass. [Technologist trial and native-speaker ratings.] Limits: replay not trial, synthetic session lengths.

6. Safety & Responsible AI

People decide. RAD-SMART only proposes. A doctor approves each course and its time; a technologist or doctor approves every change on the day, including every breakdown response. The approver and time are recorded. It never decides whether or how a patient is treated, and it is not a diagnostic or treatment-planning tool.

Rules are guaranteed, not hoped for. The department's hard rules are constraints in the optimiser, so a plan that breaks one cannot be produced; an independent checker re-verifies every plan, and the Today page shows a live rule check.

The language models are fenced in.

- They never schedule. They translate and explain decisions already made.
- **Every translation is checked**: if it drops, changes or adds a clock time, it is thrown away and the patient gets the English text. Messages name the actual date, not "next working day".
- **Every explanation is checked**: an answer that mentions a time not in the engine's facts is rejected, and the engine's own reasons are shown instead. The prompt forbids medical advice.
- **If Sarvam is unreachable or out of credit**, messages go in English and explanations show the engine's reasons. Scheduling never depends on Sarvam.
- Staff see both the English and the translated text, and can listen to the voice note, before sending.

Privacy.

- **No names, hospital numbers or phone numbers are sent to Sarvam**, only the message text with times and dates.
- The patient page shows the first name only and needs a long random link; it allows at most six replies a day, and the check-in QR code holds only a six-letter code.
- Messages are sent from the department's own WhatsApp by a person; no third-party gateway sees patient data in this phase. Telegram is used only in the online demonstration, with synthetic patients.

- All demo and evaluation data are synthetic. The department's records were anonymised before sharing and are used only as aggregates; no row-level data is published.
- The pilot follows the DPDP Act 2023 and Rules 2025 (a patient notice and consent for messaging), and goes to the Institutional Ethics Committee as a quality-improvement study.

Fairness. Older patients, patients on public transport or the hospice bus, and patients with time windows are protected by explicit soft goals or hard constraints, not left to chance. The patient page and voice notes serve patients who do not read English, and those who read little.

Fallback. If a hard rule is broken in practice during the pilot, the department returns to manual scheduling.

> **Short version.** A person approves every course and every change; RAD-SMART decides only when, never whether or how, and is not diagnostic. Hard rules are optimiser constraints, re-checked independently. Language models never schedule: a translation that changes a time is discarded (the patient gets English), an explanation that invents a time is rejected, and without Sarvam everything still works in English. No names or numbers go to Sarvam; the patient page shows the first name only; data are synthetic or aggregated; DPDP-compliant consent and ethics review for the pilot.

7. Deployment & Sustainability

Running today: live on Microsoft Azure

RAD-SMART is deployed, not just demonstrated on a laptop. The whole application ships as **one Docker image** and runs on **Microsoft Azure** at https://radsmart.braveglacier-5f9807f7.uaenorth.azurecontainerapps.io/try/.

Layer	What runs
Container	One Docker image: CARE's own production build at a pinned version, the RAD-SMART backend and frontend plugins, nginx and gunicorn; built reproducibly from a clean source export, with no secrets inside
Hosting	Azure Container Apps: HTTPS, health checks, scales to zero when idle and back up on the first visit
Database	Azure Database for PostgreSQL (Flexible Server), with automatic daily backups kept for 7 days
Images	Azure Container Registry; every release is a tagged image and the last three are kept, so a rollback is one command
Secrets	Container Apps secrets; the Sarvam key and database password are never in the code or the image
Scheduled job	A daily Container Apps job runs database migrations and renews the synthetic demonstration day
Cost and safety guards	Limits on Sarvam spending per day; a guard that takes the server offline before its free hours run out
Operations	One script (`deploy/azure/deploy.sh`) creates, builds, migrates, deploys and checks the whole system

The online server runs in **demonstration mode**: synthetic patients, one-click roles without a password, and a fresh day every morning. Switching to the pilot is configuration, not new code: demonstration mode off, staff sign in with their own CARE accounts, and the database is kept.

For the pilot, the same image runs either on the hospital's own server (Docker Compose, beside its CARE) or in an Indian Azure region such as Central India, so patient data stays in India under the DPDP Act. The hospital's IT team chooses; nothing in RAD-SMART changes.

Requirements

- A CARE instance: the department's own, a hospital VM running CARE with Docker Compose, or our ready-made Docker image on Azure. RAD-SMART installs as a backend and a frontend plugin, with no change to CARE's code.

- A laptop or tablet at the machine console and one at reception (a USB barcode scanner or the camera reads the QR code).
- The department's WhatsApp on a staff phone or WhatsApp Web.
- Internet access for Sarvam, and a Sarvam API key. Everything except translation, voice and explanation works offline.

Costs (indicative)

Item	Estimate (₹)
Server with UPS (or an existing hospital VM at no cost)	0 – 1,50,000
Or cloud hosting instead of a server: Azure Central India, one always-on container (0.5 vCPU) and a small managed database	about 4,000 – 6,000 a month
Tablets for reception and the console (or existing PCs)	0 – 60,000
QR scanner, labels	5,000 – 15,000
Sarvam API, per course (about 3 messages of 190 characters, translated at ₹20 and voiced at ₹30 per 10,000 characters, plus explanations at under ₹0.10)	**about ₹3–5**
Sarvam API, per year (about 1,000 courses, plus breakdown messages)	**about ₹5,000 – 10,000**
WhatsApp Business API, when added (utility messages)	[confirm current rate card]

The software is open source and has no licence fee. Prices are Sarvam's published list prices (Translate ₹20 and Bulbul v3 ₹30 per 10,000 characters; Sarvam-105B ₹4 input and ₹16 output per million tokens) and should be confirmed before the pilot.

Integration

RAD-SMART uses CARE's own patients, facilities, users and permissions, and adds its own tables. It is installed and removed like any CARE plugin. The next integration is a read-only feed from the oncology information system (MOSAIQ or ARIA, HL7 v2 or FHIR R4), so that courses need not be typed twice.

Path to adoption

1. **Pilot, 90 days, on the Versa HD**: two weeks measuring today's waits with the day board, three weeks in shadow mode (RAD-SMART proposes, the department keeps scheduling by hand, and we compare), then eight weeks live. Target: a 90th-percentile wait at least 25% below the recorded 109 minutes, set below our replay results.
2. **The second machine** (2027): the engine already forecasts capacity for two machines; live planning extends to it.
3. **Other CARE hospitals**: CARE is used by government and charitable hospitals across India, so a plugin that works here can be installed there. 69% of Indian radiotherapy centres have one machine, the case RAD-SMART is built for.

Sustainability. Running costs are a few thousand rupees a year. The plugin is maintained in the open with the Open Healthcare Network community, and the department's own staff can run it after the pilot.

> **Short version.** Running today on Microsoft Azure as one Docker image (CARE's production build with both plugins, managed PostgreSQL with daily backups, HTTPS, one-script deployment), in demonstration mode; the pilot is the same image with demonstration mode off, on a hospital server or in an Indian Azure region. Needs a CARE instance (installed as a plugin, no change to CARE), a console and reception device, the department's WhatsApp, and internet for Sarvam (everything else works offline). Costs: hardware ₹0–2.2 lakh (existing PCs or VM cost nothing); Sarvam about ₹3–5 per course, about ₹5,000–10,000 a year; open source,

> no licence fee. Next integration: a read-only OIS feed. Adoption: a 90-day pilot (baseline, shadow, live, target −25%), then the second machine in 2027 and other CARE hospitals.

8. Next Steps

When	Step
By 8 Nov 2026	Technologist trial of the demo; native-speaker rating of every translation
Nov–Dec 2026	Head of department approval; ethics committee review; hospital IT sign-off; patient notice and consent for messaging
Jan 2027	Pilot phase 1: two weeks measuring today's waits with the day board
Feb 2027	Pilot phase 2: three weeks in shadow mode
Feb–Apr 2027	Pilot phase 3: eight weeks live; retrain the session-minute model on real minutes
2027	WhatsApp Business API (automatic sending and replies); spoken replies; the second machine; a read-only OIS feed; offer the plugin to other CARE hospitals

> **Short version.** Before 8 Nov: technologist trial and native-speaker ratings. Nov–Dec: approvals, ethics review, consent. Jan–Apr 2027: a 90-day pilot (baseline, shadow, live) and retraining on real minutes. Later in 2027: automatic WhatsApp, spoken replies, the second machine, an OIS feed, and other CARE hospitals.

Evidence

Evidence	Where
The submission in the official template (15 slides)	`docs/RAD-SMART_Phase2_Submission.pptx` (built by `tools/build_submission.py`)
The app, live on Microsoft Azure (synthetic patients, one click per role)	radsmart...azurecontainerapps.io/try/
How it is deployed (Docker, Azure, one script)	`deploy/azure/README.md`
Project website, with the demo video	psraghothamrao8.github.io/RAD-SMART
Demo video (the real app, narrated, synthetic patients)	Website, demo section · `docs/media/RAD-SMART_demo.mp4`
Code: engine, CARE plugins, evaluation scripts	github.com/psraghothamrao8/RAD-SMART
Pitch deck	Website, deck
Research paper (method and results)	`docs/RAD-SMART_Research_Paper.pdf`
With/without impact analysis	`docs/RAD-SMART_Impact_Analysis.pdf`
Sarvam AI evaluation	`docs/evaluation/sarvam_evaluation.md`
User guide and demo script	`docs/RAD-SMART_User_Guide.pdf`
Full research report	`docs/RAD-SMART_Research_Report.pdf`
Automated tests	`app/care_radsmart/care_radsmart/tests/` (37 tests) · `poc/tests/` (8 tests, run by GitHub on every change)