OTAMS Deliverable 1 — 4-Way Team Split & Submission Checklist

Use this page to track ownership, acceptance criteria, PRs, and demo flow for **Deliverable 1**.

Student A — Auth & Data (Owner of login/logout, admin seeding)

Build

- Domain models: **User** (base), **Student**, **Tutor**, **Role**.
- **AuthRepository** (in-memory now; swappable to Firebase later) with:
- `registerStudent`, `registerTutor`, `login(email, password) → role`, `logout()`.
- **Seed Administrator** credentials (from README).
- Implement **login** → returns role; **logout**.
- *(Optional)* Simple file/JSON persistence **or** Firebase (+5% bonus).

Acceptance Criteria

- Admin can sign in with README creds; **logout** returns to **Login**.
- `login(...)` returns the **exact role string** consumed by Welcome screen.

PRs to Open

- `data/` (repository + models), `auth/` (use-cases), **admin seeding**.

Student B — Student Registration + Validation

Build

- **Register Student** screen with fields: `firstName`, `lastName`, `email`, `password`, `phone`, `programOfStudy`.
- Full **field validation** with inline errors; **no submit** until valid.
- On success: call repo → toast **"Registered, please log in."**

Acceptance Criteria

- Cannot submit with blank/invalid fields; **clear messages** shown.
- Student account is created via repository.

PRs to Open

- `ui/register/student/*`, `validation/*`, tests or preview screenshots.

Student C — Tutor Registration (+ Multi-course) + Validation

Build

- **Register Tutor** screen with fields: `firstName`, `lastName`, `email`, `password`, `phone`, `highestDegree`, `coursesOffered` (**one or more**) with add/remove UI.
- Full **field validation** (must include **≥1 course**).
- On success: call repo → toast.

Acceptance Criteria

- Multiple courses can be added; errors appear for invalid/empty state.
- Tutor account created via repository.

PRs to Open

- `ui/register/tutor/*`, `components/CourseChips/*`, validation helpers.

Student D — App Shell, Welcome Screen, and Submission Artefacts

Build

- **Navigation:** Login → Register Student/Tutor → Login → **Welcome**.
- **Welcome Screen** prints **exact text**:

`Welcome! You are logged in as <role>`
(Student/Tutor/Administrator)

- **Log off** button → back to **Login**.
- **UML class diagram (PDF)** of **domain model only** (no Activities/UI classes).
- **README** with **Admin credentials**.
- **Demo video** (≤ 5 min) showing rubric items.
- **GitHub Release:** tag **v0.1**, title **Deliverable1**; attach **APK (renamed)**, **video**, **UML**, **README**.

Acceptance Criteria

- Exact welcome string appears for **each role**.
- Release has **correct tag/title** and **attachments**; APK **renamed** per team.

PRs to Open

- `ui/welcome/*`, `nav/*`, `README.md`, `/docs/UML_v0.1.pdf`, **release checklist**.

Cross-cutting: Validation & Marking Coverage

- **Validation (20%)**: B and C implement per field; A/D verify at login & welcome.
- **Create accounts (15%)**: B (Student) + C (Tutor).
- **Welcome with role (15%)**: D consumes A's role.
- **Log off (10%)**: A (logic) + D (UI).
- **Team/commit checks (10% + 10%)**: D is accountable to verify; **all four** appear in Classroom team and **each makes ≥1 commit**.
- **UML (5%)**, **APK submission (5%)**, **video (10%)**: D owns; others review.

RACI & Reviews (Keep Everyone Contributing)

- **R (Responsible):** Owners listed per section above.
- **A (Accountable):**
- A for **auth/data**
- B for **Student form**
- C for **Tutor form**
- D for **submission**
- **C/I (Consulted/Informed):** Everyone reviews each other's PRs.

Review Plan

- **A** reviews **B & C** (field rules align with repository schema & role handling).
- **B** reviews **C** (courses UI & validation).
- **C** reviews **B** (student form UX & validation).
- **D** reviews **all** for **exact text** & **rubric match**, then records video.

Demo Flow (Script — each person records their part)

1) **B:** Register **Student** (show a validation edge case, then success). 2) **C:** Register **Tutor** (add **multiple courses**; show validation, then success). 3) **A:** Login as **Admin** (from README) → **Log off**. 4) **A/D:** Login as **Student** → Welcome (role shown) → Log off; Login as **Tutor** → Welcome → Log off. 5) **D:** Show **Release** page with **v0.1**, **Deliverable1**, **renamed APK**, **UML PDF**, **README**, **video** attached.

Final Hand-In Checklist (Tick Before Submit)

- [ ] Admin creds in **README**; admin seeded in app
- [ ] Student form: all fields + validation + repo call
- [ ] Tutor form: all fields + **multi-course** + validation + repo call
- [ ] Login returns exact **role**; Welcome shows exact text
- [ ] **Log off** returns to Login (no back-stack leaks)
- [ ] **UML PDF** (domain-only) exported to `/docs/UML_v0.1.pdf`

- [ ] GitHub team has **all four members**; each made **≥1 commit**
- [ ] **Release** `v0.1` titled **Deliverable1** with **renamed APK**, **video**, **UML**, **README**
- [ ] *(Optional)* Firebase Auth + Firestore users implemented (+5%)

Notes

- Keep Welcome text **exact** (markers look for it).
- Rename the APK exactly as instructed (e.g., `Project_Group_X_debug_apk`).
- Use small, frequent commits so everyone appears in the history.