Assumptions & Edge Cases

Req1 — Account creation & verification

Requirement (given): Anyone can create an account with a unique, valid email and strong password; account types: student, faculty, staff, partner; **university accounts require verification**.

Assumptions

1. **Email uniqueness** is enforced system-wide (case-insensitive).
2. **Password policy** = at least one uppercase, lowercase, digit, symbol; min length 8 (configurable).
3. **University verification**: emails ending **@yorku.ca / @my.yorku.ca** trigger a verification step (internal check; no separate actor in UCD).
4. **Partners** may use any valid email; **no university verification** required.
5. Account type chosen at registration; can be changed only by **Admin** (not required to implement in D1).

Edge cases / acceptance

- Duplicate email → registration rejected with guidance.
- Weak password → rejected with policy hint.
- Verification must complete before booking is allowed for university accounts.

Req2 — Chief Event Coordinator creates admins

Requirement (given): Only the **Chief Event Coordinator** can auto-generate admin accounts for room management.

Assumptions

1. There is **exactly one** Chief account at a time.
2. **Admins cannot create other admins** (no delegation).
3. New admin credentials (AdminID, temp password) are auto-generated and must be changed on first login.

Edge cases / acceptance

- Attempt by non-Chief to create admin → rejected.
- Chief can view current admins list (view only; not mandated to delete in D1).

Req3 — Registered users can book; hourly rates vary by type

Requirement (given): Users can **book available rooms**; hourly rates: Student $20, Faculty $30, Staff $40, Partner $50.

Assumptions

1. A **Rate Table** maps user type → hourly rate; rates applied at **booking time**.
2. **Availability** is recomputed upon each create/modify/cancel/extend operation.
3. Booking requires **start time, end time, room**, and the user's **account type** (from profile).

Edge cases / acceptance

- Overlapping request for the **same room** and time is **rejected** (internal validation, not a new requirement).
- Minimum booking granularity = **30 minutes** (configurable). *(Assumption for modeling; not required by spec.)*

Req4 — One-hour fee upfront; 30-minute no-show rule

Requirement (given): Collect **1 hour fee upfront**. If no check-in **within 30 min of start**, **deposit is forfeited**; otherwise **applied** to final cost.

Assumptions

1. The upfront charge is treated as a **deposit/hold** equal to **one hour** at the user's rate.
2. **Auto-cancel** after 30 minutes of no valid check-in and **free the room** for others. *(Explicit assumption; spec only mandates forfeiture.)*
3. **Scheduler** triggers the 30-minute check; **Sensor/Badge** event confirms check-in.

Edge cases / acceptance

- Late arrival after 30 min → deposit forfeited; booking (per A2) is canceled and room becomes available.
- Early arrival (before start) does **not** start the timer; check-in is valid only within policy your team sets (e.g., ±10 min). *(Assumption; optional to state in activity diagram.)*

Req5 — Sensors detect occupancy; badge scan verification; data sent to system

Requirement (given): Each room has sensors for **occupancy + badge**; data flows into the system.

Assumptions

1. Sensor gateway posts data at a **configurable interval** (default 60s).
2. A **badge match** for the booked user within the room and time window counts as **check-in success**.
3. Occupancy without valid badge does **not** count as check-in (may be flagged for admin review—informational only in D1).

Edge cases / acceptance

- Multiple badges for a booking (group) → only the **booking owner's** badge is required. *(Assumption to keep scope simple.)*

Req6 — Admins add/enable/disable rooms; temporary closures for maintenance

Requirement (given): Admins can **add/enable/disable rooms**; rooms can be **closed temporarily** for repairs/maintenance.

Assumptions

1. **Disable** hides the room from search/booking; existing active bookings remain unless they overlap with **maintenance**.

2. **Maintenance windows** have start–end times; overlapping bookings are flagged for **admin manual resolution** in D1 (inform, not auto-refund).

Edge cases / acceptance

- Attempt to book a disabled/maintenance room → rejected.

Req7 — Room details: unique ID, capacity, building/room location

Requirement (given): Rooms have **unique ID, capacity, building/room location**.
Assumptions

1. **RoomID** is unique and stable.

2. Optional metadata (e.g., equipment) may exist but is **not required** by Req7.

Edge cases / acceptance

- Admin can update room fields **except** RoomID during an **active booking** window. *(Assumption for consistency.)*

Req8 — User provides org ID/student #; can edit/cancel before start

Requirement (given): Users must provide **org ID/student number**; bookings can be **edited or canceled before start**.

Assumptions

1. System validates **format** (not existence) of the provided ID.
2. **Edit** may change time/room; must **re-validate availability** (internal to Req3/9 flows).
3. **Cancel** before start reverses/voids any non-finalized charges (deposit handling per payment gateway rules).

Edge cases / acceptance

- Attempt to edit/cancel **after start** → rejected (unless using **Extend** per Req9).

Req9 — Bookings may be extended before expiry if available

Requirement (given): Users can **extend before expiry** if the room is **available**.

Assumptions

1. "Expiry" = the **scheduled end time** of the current booking.
2. An **extend** request is allowed up to a small buffer (e.g., **≤10 min before end**) and only if the **following slot** is free.
3. **Extension charge** is collected immediately (same method as original or prompted).

Edge cases / acceptance

- If the next slot is partially free, extension is **not** allowed (keep to full slot increments per your time granularity).

Req10 — Payments: credit, debit, institutional billing

Requirement (given): Support **credit**, **debit**, and **institutional billing**.

Assumptions

1. **Only** these three are in scope for D1 (no Apple/Google Pay, PayPal).
2. Payment responses: **Approved / Declined / Pending**; **only Approved** confirms the booking.
3. **Deposit capture** at booking; **final settlement** at checkout (or after use) per your activity/sequence diagrams.

Edge cases / acceptance

- Deposit Approved but Final Payment Declined → booking stands as completed usage; balance due is **out of D1 scope** (note in assumptions).