Test Project

Module C – Global Events

Web Technologies

Submitted by:

Lifeng Song, Zhonghui Group, Independent Test Project Designer
Zhaozhen Wang, Zhonghui Group, Independent Test Project Designer
Changxu Dong, Zhonghui Group, Independent Test Project Designer
Xiaoqiang Wei - Zhonghui Group, Independent Test Project Designer
(Thomas) Seng Hin MAK, Independent Test Project Designer

Gilles Granger, Indepdendent Test Project Validator
Hok Kin Fong, Indepdendent Test Project Validiator

Contents

Introduction **3**
Description of project and tasks **3**
Model and Relationships 3
Authentication for staffs 4
Events Management (Top Level) 4
Rooms Management (under an event) 4
Speakers Management (top-level, multi-event) 4
Sessions Management (under an event) 5
Reservations / Registrations (under an event) 5
Public JSON Capacity API (no authentication) 5
Audit Logs (reservations only, under the event) 6
Automated Testing Specifications 6
Instructions to the Competitor **7**
Implementation Checklist 7

Introduction

An intranet-only, server-side staff application to manage global events and their nested resources (rooms, sessions, ticket capacities, reservations), plus a global speakers directory. The system must enforce session double-booking rules, concurrency-safe ticket capacity, and an audit trail for reservation create / update / cancel.

- Server-side rendered application with server-side session management.
- Staff authenticate with an employee number only (no password). The value is digits-only (numeric barcode payload). Entering it simulates scanning an employee ID barcode on a staff badge.
- All staff have the same permissions (no role hierarchy).
- Top-level navigation after login: Events, Speakers, plus Logout (and Login when logged out). Rooms, sessions, ticket capacities, reservations, and the audit log for that event are managed under a selected event (not as top-level nav items).
- Information architecture (required outcome): the product is event-centric. Staff must open an event before managing that event's rooms, sessions, ticket capacities, reservations, or reservation audit log. Do not present Rooms/Sessions/Reservations as global top-level areas parallel to Events (except Speakers, which are global and many-to-many with events).
- Audit logging is required only for reservation create, update, and cancel. Other CRUD does not need audit entries.
- The application must be responsive and accessible.

Description of project and tasks

- The database must be designed by the competitor in Third Normal Form (3NF).
- Physical foreign key constraints and correct column types must be present.
- Please produce db-dump.sql with schema, FKs, seed data (including the three fixed employee numbers below and sample events).

Model and Relationships

- Almost all operational data is under an Event. Exception: Speakers are global and link to many Events (many-to-many), because the same speaker may appear at multiple events.
- Event has many Rooms; Room belongs to one Event.
- Event has many Sessions; Session belongs to one Event and optionally one Room of that event (room may be null).
- Event ↔ Speakers: many-to-many (e.g. event_speaker). Session ↔ Speakers: many-to-many (e.g. session_speaker).
- Ticket types are fixed (hard-coded) as three codes: early_bird (Early Bird), normal (Normal), vip (VIP). No need to implement arbitrary ticket-type CRUD.
- Each Event stores its own capacity for each of the three ticket types (editable in the event back office). Remaining seats = capacity – sum of reserved quantities for that type, counting only active (non-cancelled) reservations.
- Event has many Reservations. A reservation belongs to one Event and records who is booked and how many seats of each ticket type. A reservation has a status such as active or cancelled; only active reservations consume capacity.
- Each Event has an independent time zone (IANA id, e.g. Asia/Shanghai, Europe/Berlin, America/New_York). You may find the IANA ids in media assets.

© WorldSkills International

- Session start/end must be stored in the database as UTC instants (UTC DATETIME/TIMESTAMP, or an equivalent documented UTC convention).
- Staff UI for creating, editing, and listing sessions must enter and display times in the parent event's time zone (not raw UTC unless the event TZ is UTC).
- Assessment example: event timezone Asia/Shanghai (UTC+8, no DST). Staff enters local start 2026-09-22 09:00 → database stores 2026-09-22 01:00:00 UTC (or equivalent). Session list/detail shows the local time again for that event.
- No need to handle time zone changing for event after it is created.

Authentication for staffs

- Accessing protected back-office routes without authentication must return HTTP 401 and redirect to the login page.
- Login form: employee number only (barcode scan simulation). Values are pure digits (no letters), matching a typical numeric barcode printed on the staff badge. Store and compare as a string of digits.
- The following three employee numbers are fixed for the competition and must be accepted as valid staff logins (seed them in the database regardless of schema choices):
  - 1001001
  - 1001002
  - 1001003
- Invalid employee numbers show an error in the error-message container.
- After login, persistent top navigation shows Events, Speakers, and Logout.

Events Management (Top Level)

- Staff can create, list, view, update, and delete events.
- Event fields must include at least: name/title, and time zone (selectable or enterable IANA id). Other optional fields that a staff can input: location/city, description.
- From an event detail (or event hub) page, staff can manage that event's Rooms, Sessions, ticket capacities, Reservations, and the event's Audit log.
- Per-event ticket capacities: staff can edit the capacity integers for Early Bird, Normal, and VIP. Capacity cannot be set below the number of seats already reserved for that type (reject with clear error / HTTP 422).

Rooms Management (under an event)

- Staff can create, list, update, and delete rooms belonging to the selected event.
- Room fields must include at least a name.
- Deleting a room is allowed even if sessions still reference it. On delete, every session that used that room must have its room set to empty/null (session remains; staff can assign another room later).

Speakers Management (top-level, multi-event)

- Staff can create, list, update, and delete speakers at the global Speakers area (top nav).
- Add speaker form accessible via /speakers/new.
- Speaker profile includes biography and profile picture.
- Profile picture: validate real MIME type (not extension only). Allowed: image/jpeg, image/png, image/webp. Max size 2 MB.
- Speakers may be linked to multiple events (many-to-many).

- Deleting a speaker is allowed. After delete, any session_speaker (or equivalent) links for that speaker must be removed so sessions no longer reference the deleted speaker (session remains; speaker list on that session becomes empty or without that speaker).

Sessions Management (under an event)

- Staff can create, list, update, and delete sessions for the selected event.
- Each session belongs to exactly one event. A session may have a room of that event, or no room (null) after a room was deleted or if not yet assigned.
- Session start/end: store UTC; staff enter/display in the parent event's timezone (see section 3). Absolute time must round-trip correctly (local → UTC → local).
- End must be strictly after start (compare as absolute instants); otherwise HTTP 422 from server with a clear error.
- When a room is set, reject save if that room is already used by another session of the same event with an overlapping time range. Sessions with null room skip room-overlap checks.
- Reject save if any assigned speaker is already assigned to another overlapping session. (Only handle speaker's time conflict within an event. No need to handle or check any speaker's time conflict across events)
- Staff can assign zero or more speakers to a session.

Reservations/Registrations (under an event)

- Under an event, staff can list all reservations for that event.
- Staff can create a reservation for a person: attendee full name, email, and quantities of tickets by type (how many Early Bird, how many Normal, how many VIP). At least one quantity must be ≥ 1. Default UI may preselect Normal quantity 1 and others 0.
- Name and email are both required (non-empty after trim); email must be a valid format. Email uniqueness is not required.
- Staff can update a reservation (name, email, and/or quantities per ticket type).
- Staff can cancel a reservation (e.g. set status to cancelled). Cancelled seats must no longer count toward reserved totals so capacity is freed for future bookings. Cancelled rows may remain visible in the list for history.
- Cancelled reservation cannot be re-active.
- Saving create/update must fail if any ticket type's resulting reserved total (active only) would exceed that type's capacity for the event (HTTP 422, clear message in error-message).
- On reservation forms/UI that offer ticket types, types with remaining 0 are disabled and marked "Sold Out" (data-testid="ticket-sold-out" on sold-out options as applicable). When editing an existing reservation, quantities already held by that reservation may still be editable even if remaining appears 0 for new sales. But editing an existing reservation still cannot increase the quantities over capacities.

Public JSON Capacity API (no authentication)

A public read-only JSON API (no login) exposes remaining ticket capacity. Use these contracts exactly:

- GET /api/events — list events with capacity summary.

```
[
  {
    "id": 1,
    "name": "WorldSkills Shanghai 2026 Opening",
    "timezone": "Asia/Shanghai",
    "tickets": {
```

```
    "early_bird": { "capacity": 100, "reserved": 40, "remaining": 60 },
    "normal":    { "capacity": 200, "reserved": 200, "remaining": 0 },
    "vip":       { "capacity": 50,  "reserved": 10, "remaining": 40 }
   }
  }
]
```

- GET /api/events/{id} — single event, same ticket object shape as one array element above. HTTP 404 if missing.
- Keys early_bird, normal, vip are required. remaining must equal capacity – reserved. Response Content-Type: application/json.
- This API does not require authentication (usable by a public kiosk or partner site). It is read-only; mutations stay in the staff UI.

Audit Logs (reservations only, under the event)

- Write an audit log row for every reservation create, update, and cancel. No other entity CRUD requires audit logging.
- Each audit entry must store at least:
  - operator_employee_number — the logged-in staff number (1001001 / 1001002 / 1001003)
  - action — one of: create, update, cancel
  - entity_type — e.g. reservation
  - entity_id — primary key of the reservation
  - event_id — parent event
  - created_at — timestamp (UTC)
  - summary — human-readable snapshot including attendee name, email, and quantities per ticket type
- For update: include before and after summary (or structured before_json / after_json) so reviewers can see what changed.
- For cancel: summary must show the cancelled name, email, and quantities released.
- The audit log UI is under the event, listing only that event's reservation audit entries—not a global top-level Logs nav item. Use data-testid="nav-event-logs" (or equivalent control on the event page with data-testid="event-audit-log").

Automated Testing Specifications

- Login employee number input must use data-testid="login-emp-num".
- Login button must use data-testid="login-btn".
- Logout control must use data-testid="logout-btn".
- Navigation — Events must use data-testid="nav-events".
- Navigation — Speakers must use data-testid="nav-speakers".
- Event name/title input must use data-testid="event-name".
- Event timezone input/select must use data-testid="event-timezone".
- Event submit button must use data-testid="event-submit-btn".
- Event Early Bird capacity input must use data-testid="event-cap-early-bird".
- Event Normal capacity input must use data-testid="event-cap-normal".
- Event VIP capacity input must use data-testid="event-cap-vip".
- Event sub-nav — Audit log (under event) must use data-testid="nav-event-logs".
- Event audit log list container must use data-testid="event-audit-log".
- Room name input must use data-testid="room-name".

© WorldSkills International

- Room submit button must use data-testid="room-submit-btn".
- Session title input must use data-testid="session-title".
- Session room select must use data-testid="session-room".
- Session speaker select must use data-testid="session-speaker".
- Session start datetime must use data-testid="session-start".
- Session end datetime must use data-testid="session-end".
- Session submit button must use data-testid="session-submit-btn".
- Registration attendee name input must use data-testid="registration-name".
- Registration email input must use data-testid="registration-email".
- Registration Early Bird quantity must use data-testid="registration-qty-early-bird".
- Registration Normal quantity must use data-testid="registration-qty-normal".
- Registration VIP quantity must use data-testid="registration-qty-vip".
- Registration submit button must use data-testid="registration-submit-btn".
- Registration cancel button must use data-testid="registration-cancel-btn".
- Sold-out ticket option must use data-testid="ticket-sold-out".
- Error message container must use data-testid="error-message".

Instructions to the Competitor

- You need to commit your project to the git repository that you created on the submission platform.
- Provide db-dump.sql with schema, constraints, seed data (must include employees 1001001, 1001002, 1001003), and sample records
- Git repository with meaningful commits.
- Please access to the submission platform dashboard by using the credentials you received.
- Please include an expert_readme.txt on how to get your project up and running, and where to access your project.

Implementation Checklist

- Login via numeric employee barcode (1001001–1001003); equal permissions.
- Top nav: Events, Speakers, Logout (no global Logs); event-centric hub for nested resources.
- Event CRUD + per-event IANA timezone + per-type capacities; Room/Session/Reservation/Audit under event.
- Delete room → related sessions' room set to null; delete speaker → clear session speaker links.
- Sessions: UTC in DB; enter/display in event TZ; room/speaker overlap rules.
- Reservations: name + email + quantities; edit + cancel.
- Public GET /api/events and GET /api/events/{id} JSON as specified (no auth).
- Audit only create/update/cancel reservation; list under the event.

© WorldSkills International