OSCEBuilder: The OSCE Rotation Scheduler

Documentation & User Guide

Release Date: December 2025

Platform: Universal (HTML5/JavaScript) - Client-Side Only

Table of Contents

1) Overview

1.1 Purpose

The OSCE Rotation Scheduler is a **single-file HTML application** that helps academic teams create, validate, visualize, and print OSCE rotation schedules **offline**.

1.2 Outputs You Can Produce

- **Master Schedule Report** (overview of rotation allocations)
- **Student View Report** (tables per student, printed sequentially)
- **Room View Report** (who is in each room per time block)
- **Flow Map** and **Flow Map + Legend** (single A4 page auto-fit)

CALLOUT: Why it's different
Unlike generic online schedulers, this app is designed for OSCE logic and produces **print-grade A4 outputs** without any server.

2) System Requirements & Privacy

2.1 Recommended Hardware

- Modern laptop/desktop, 8GB RAM or more
- Screen: 1366×768 minimum (best: 1920×1080+)

2.2 Supported Browsers

- **Recommended:** Google Chrome, Microsoft Edge
- Supported: Safari, Firefox (printing/scaling behavior may differ slightly)

2.3 Offline & Data Security

- Runs fully in your browser.
- No internet required.
- No data is transmitted anywhere.
- Your data persists only if you **Save Project** (JSON file) or keep the tab open.

WARNING: Sensitive assessment data
If student lists are confidential, store saved JSON files in approved institutional locations and follow your data retention policy.

3) Getting Started (Quick Start)

3.1 Open the Application

1. Locate the application on acadexailabhub.com
2. Open it in Chrome/Edge (recommended).

3.2 Load Demo Data (optional)

Use **Demo Data** to populate sample lists (students, rooms, stations, examiners) for exploration.

3.3 Add Your Real Data

You can enter data in two ways:

- Upload CSV files (fastest)
- Fill in fields on screen (manual entry)

3.4 Generate Schedule

Click **Generate Schedule** → review outputs on-screen.

3.5 Print Reports to PDF

Choose a print mode (Master, Students, Rooms, Flow Map) → **Print PDF**.

4) Data Input Methods (CSV vs On-Screen Entry)

4.1 Method A: CSV Upload (Recommended)

You can import:

- Students
- Rooms
- Stations
- Examiners

TIP: Save a "template CSV pack"
Keep standard CSV templates for recurring OSCE formats (e.g., PharmD OSCE, clinical OSCE, communication OSCE) so each event starts with plug-and-play uploading.

CSV rules (must follow)

- First row = headers
- Use comma-separated format
- Avoid commas inside a field unless quoted

CALLOUT: Common import failure
90% of import errors come from a header mismatch (e.g., Student Id instead of StudentID) or extra commas.

4.2 Method B: On-Screen Entry (Manual)

Use on-screen forms to:

- Add rows
- Edit fields
- Remove entries

Best use cases

- Small OSCEs
- Quick testing
- Last-minute adjustments

5) Core Scheduling Concepts

5.1 Entities

- **Students:** participants rotating through stations
- **Stations:** OSCE tasks/competencies
- **Rooms:** physical locations mapped to stations
- **Examiners:** assessors assigned to rooms (if configured)

5.2 Rotation Logic

The schedule is constructed as a series of **time blocks**. In each time block:

- Each student is assigned to a room/station
- The model ensures consistent progression through the OSCE circuit

5.3 Flow Map (Rotation Diagram)

The Flow Map shows the “movement logic” of students through rooms over blocks.

TIP: Use Flow Map + Legend for faculty briefing
It’s ideal for short OSCE orientation sessions because it explains the rotation at a glance.

6) Generating the Schedule

6.1 Pre-Generation Checklist

Before clicking **Generate Schedule**, confirm:

- Students list is loaded (not empty)
- Rooms and stations are defined
- Rotation map/order is complete
- Any timing parameters are correct

6.2 Generate Schedule

1. Click **Generate Schedule**
2. Confirm success message
3. Switch to:
    - Master View
    - Student View
    - Room View
    - Flow Map / Flow Map + Legend

WARNING: “Generated successfully” but no output
This usually means required inputs are missing OR the view is filtered/empty. See Troubleshooting section 10.

7) Views & Dashboards

7.1 Master View

Shows the overall schedule structure and aggregated allocation.

Use it for

- Validating global conflicts
- Confirming station balance
- Sharing with OSCE leadership

7.2 Student View

Each student’s station sequence across time blocks.

Print behavior

- Student tables print **successively** (continuous) rather than one student per page.

7.3 Room View

Each room’s timeline—who is assigned in each block.

Use it for

- Invigilator briefings
- Room signage prep
- Station readiness checks

7.4 Flow Map View

High-level rotation diagram.

Modes

- Flow Map
- **Flow Map + Legend (A4 Fit)**: auto-scaled to fit one A4 page

8) Printing & PDF Reporting (A4 Optimized)

8.1 Printing Modes

Select from the Print Mode dropdown:

- Master View (Print)
- Student View (Print)
- Room View (Print)
- Flow Map
- **Flow Map + Legend (A4 Fit)**

8.2 Print Quality

The app prints in **sharp black** (not grey) for clarity.

TIP: Best PDF output
Use Chrome/Edge → Destination: “Save as PDF” → default scaling.

8.3 Flow Map A4 Fit Guarantee

Flow Map + Legend mode:

- Measures printable A4 height
- Auto-scales the diagram + legend
- Ensures a single-page output (where possible)

CALLOUT: If it still looks crowded
Reduce long room names or reduce the number of nodes shown in one circuit (depending on your OSCE design).

9) Save / Load (Continue where you left off)

9.1 Save Project

Click **Save Project** anytime. The app downloads a .json file that may include:

- Students, rooms, stations, examiners
- Rotation map/order and configuration
- Generated schedule results (if present)

Filename best practice

- OSCE_Schedule_YYYY-MM-DD.json
- Example: OSCE_Schedule_2025-12-02.json

9.2 Load Project

Click **Load Project** and select a previously saved JSON file. The app restores your session so you can continue.

WARNING: Don't edit the JSON manually
Manual edits commonly break file structure and cause load failures.

10) Troubleshooting (Errors, Meanings, Fixes)

10.1 CSV Import Problems

Message: "Headers not recognized / missing required columns"

- **Meaning:** CSV columns don't match expected names.
- **Fix:** Confirm exact header spelling and remove extra spaces.

Message: "Failed to parse CSV"

- **Meaning:** Extra commas, unquoted commas in a field, or wrong delimiter.
- **Fix:** Quote fields with commas and re-save as CSV.

Message: "Duplicate IDs detected"

- **Meaning:** Two records share the same identifier.
- **Fix:** Ensure IDs are unique (S001… S002…).

10.2 Schedule Generation Problems

Message: "No students provided"

- **Meaning:** Student list is empty.
- **Fix:** Upload students CSV or enter manually.

Message: "Rooms/stations not configured"

- **Meaning:** The circuit structure is incomplete.
- **Fix:** Add rooms and stations; confirm mapping.

Message: "Rotation map incomplete"

- **Meaning:** Not all rooms/stations are linked correctly in the flow order.
- **Fix:** Complete the rotation map/order for every room.

Message: "Uneven distribution / warning"

- **Meaning:** Student count doesn't divide evenly by blocks/rooms.
- **Fix options:** Add buffer station, allow idle slots, or adjust number of rooms.

10.3 Printing Problems

Problem: Blank/empty PDF page

- **Meaning:** Browser did not render print container.
- **Fix:** Use Chrome/Edge and ensure the app's print engine keeps print elements on-page (your current version includes this fix).

Problem: Flow map cut off or split

- **Meaning:** Scaling overflow.
- **Fix:** Use **Flow Map + Legend (A4 Fit)** mode.

Problem: Student tables print one per page

- **Meaning:** Page-break rules enabled.
- **Fix:** Use Student View "successive printing" mode; avoid per-student page breaks.

10.4 Save/Load Problems

Message: "Invalid JSON file"

- **Meaning:** Wrong file type or corrupted file.
- **Fix:** Load only JSON files produced by "Save Project".

Load succeeds but nothing appears

- **Meaning:** UI not refreshed or browser blocked local file flow.
- **Fix:** Reopen app in Chrome/Edge, then load again.

11) Best-Practice Workflow Checklist (Operational)

Before OSCE planning

- Build standard CSV templates (Students, Rooms, Stations, Examiners)
- Create a reusable "rotation map template" JSON for recurring OSCEs

During schedule build

- Upload data and confirm counts
- Generate schedule
- Validate room and student views
- Print Flow Map + Legend for briefing

Before final release

- Save final JSON to repository/archive
- Export PDFs (Master + Rooms + Students + Flow Map)
- Distribute PDFs via approved channels

12) Appendix A — CSV Templates (Copy/Paste)

A1. Students

```
StudentID,StudentName
S001,Student One
S002,Student Two
```

A2. Rooms

```
RoomID,RoomName
R01,Room 101
R02,Room 102
```

A3. Stations

```
StationID,StationName,DurationMin
ST01,Station 1,8
ST02,Station 2,8
```

A4. Examiners

```
ExaminerID,ExaminerName
E01,Examiner One
E02,Examiner Two
```

13) Appendix B — Glossary

- **Circuit:** The complete set of stations in one OSCE round.
- **Time block:** A scheduling interval where each student occupies one station/room.
- **Flow map:** Visual representation of rotation order across rooms.
- **Master view:** Highest-level schedule summary.
- **Student view:** Individual progression table for each student.
- **Room view:** Timeline of occupants per room.

14) Technical Specifications

- **Architecture:** Single-Page Application (SPA).
- **Language:** HTML5, CSS3, Vanilla JavaScript (ES6).
- **Dependencies:** Chart.js (via CDN) for visualization.
- **Privacy:** No server-side processing. No cookies. Local Storage is not used (data persists only in RAM or via manual JSON save).
- **Browser Compatibility:** Chrome, Edge, Firefox, Safari (Desktop & Mobile).
- **Concept & Logic:** Dr. Muhammad AlShorbagy, Dean, College of Pharmacy, GMU.

- **Technical Implementation:** AI-Assisted Development (Code generation).
- **Methodology:** "This single-file HTML application demonstrates a 'No-Code/Low-Code' development approach. The domain expertise, algorithm logic, and user experience design were provided by Dr. Muhammad AlShorbagy, while the source code was generated via prompt engineering using Large Language Models (LLMs)."