

Assignment 1: Mandatory

Create a Web Page and Display Scraped Event Data Using Open-Source Tools

Objective

Build a website that **lists all events in a specific city (Sydney, Australia)**. Events should be **automatically scraped** from public event websites and displayed in a **minimal, beautiful, user-friendly UI**.

Deadline: Day after tomorrow EOD

A) Event Scraping + Auto Updates

1. **Automatically scrape events** from multiple event websites for **Sydney, Australia**.
2. Store scraped events in a database with relevant fields (e.g.):
 - Title
 - Date & time
 - Venue name + address (if available)
 - City
 - Description / short summary
 - Category / tags (if available)
 - Image / poster URL
 - Source website name
 - Original event URL
 - Last scraped time
3. Events on your website should **update automatically** as they are **published/updated/removed** on the original sites:
 - Detect **new events**
 - Detect **updated events** (changed time/venue/details)
 - Detect **inactive events** (no longer available / removed / past cutoff)

You may use any open-source tools/libraries/frameworks.

B) Event Listing Website

1. Display events in a **minimalistic UI** (clean, modern, readable).
 2. Each event card should show:
 - Event name
 - Date/time
 - Venue (or location info)
 - Small description/summary
 - Source
 - “GET TICKETS” CTA
 3. Clicking **GET TICKETS** should:
 - Ask for the user's **email address**
 - Include an **email opt-in checkbox** (consent)
 - Save email + consent + event reference in DB
 - Then **redirect to the original event URL**
-

C) Google OAuth + Dashboard (Added Requirement)

Add **Google OAuth login** and a **basic admin-style dashboard** to demonstrate end-to-end MERN capability.

1) Authentication

- Google OAuth sign-in
- Only logged-in users can access the dashboard

2) Dashboard Features

A minimal dashboard with:

Filters

- City filter (default: Sydney; allow scalable multi-city)
- Keyword search (title/venue/description)
- Date range filter

Views

- **Table view** of events (rows with key fields)
- A **preview panel** (click a row → show full details on the side)

Actions

- “Import to platform” button per event
 - Sets an `imported` status
 - Stores an `importedAt`, `importedBy`, and optional `importNotes`

Status Tags

Each event must have status tags:

- `new` (freshly discovered)
- `updated` (changed since last scrape)
- `inactive` (removed from source / expired / not valid anymore)
- `imported` (imported into platform)

You should demonstrate the full pipeline: scrape → store → display → review → import → tag updates.

Assignment 2: Optional

Build an MVP: Event Recommendation Assistant (Open-Source LLM)

Create a trained MVP where an open-source LLM helps users pick events based on preferences, and notifies them when matching events appear in Sydney.

Requirements

1. User interacts via **text chat** (WhatsApp/Telegram acceptable).
2. Collect preferences such as:
 - Music/genre, budget, date/time preference, location, crowd type, etc.
3. Recommend events from your scraped DB.
4. Notify user when a matching event is added/updated.

Hint

- Use LangChain (or similar) for orchestration
- Use an open-source LLM (e.g. Llama-family / Mistral-family)
- Use simple vector search for matching + preferences memory

Submission Requirements

1. A **functional live web page link** (active link for review).
2. A **brief report (1–2 pages)**