Louder World Internship Assignment – Final Report

# Objective

The objective of this assignment was to develop a web-based application that automatically scrapes live event listings from Sydney and displays them in a clean, minimalistic format. Additionally, the application should capture user email addresses for ticket redirection purposes.

# Tech Stack

- Python 3.11  
- Django 5.2  
- Django REST Framework  
- BeautifulSoup  
- Bootstrap 5 (CDN)  
- SQLite (default Django database)

# Approach

The backend was implemented using Django, with REST APIs exposed using Django REST Framework. A Python-based scraper using BeautifulSoup pulls event data from Timeout.com. Events are saved to the database and displayed on a Bootstrap-enhanced frontend using Django templates. Users can click a 'GET TICKETS' button, which opens a modal to capture their email and then redirects them to the original event page. Captured emails are stored in a separate database model. Event data is refreshed automatically using a custom Django management command scheduled through Task Scheduler.

# Key Features

- Live scraping of Sydney events from Timeout.com  
- Modern UI with Bootstrap 5 and card-based layout  
- Email capture modal before redirection  
- Responsive frontend layout  
- API endpoint for event data using Django REST Framework  
- Automatic event updates via scheduled task

# Challenges Faced

- Timeout.com uses dynamic class names, requiring flexible selectors  
- Some event descriptions are only accessible on detail pages  
- Designing a minimal UI while ensuring all features are clear and functional  
- Managing automation via Task Scheduler with virtual environments

# Possible Improvements

- Enhance scraper to retrieve full event descriptions reliably  
- Add event images for visual engagement  
- Deploy on a public platform (e.g., Render or Vercel)  
- Implement advanced filtering and search  
- Add email validation and optional OTP

# Conclusion

All assignment requirements were successfully met, including automated scraping, clean frontend design, email capture with redirection, and scheduled data refresh. The application is modular, maintainable, and scalable for future use.