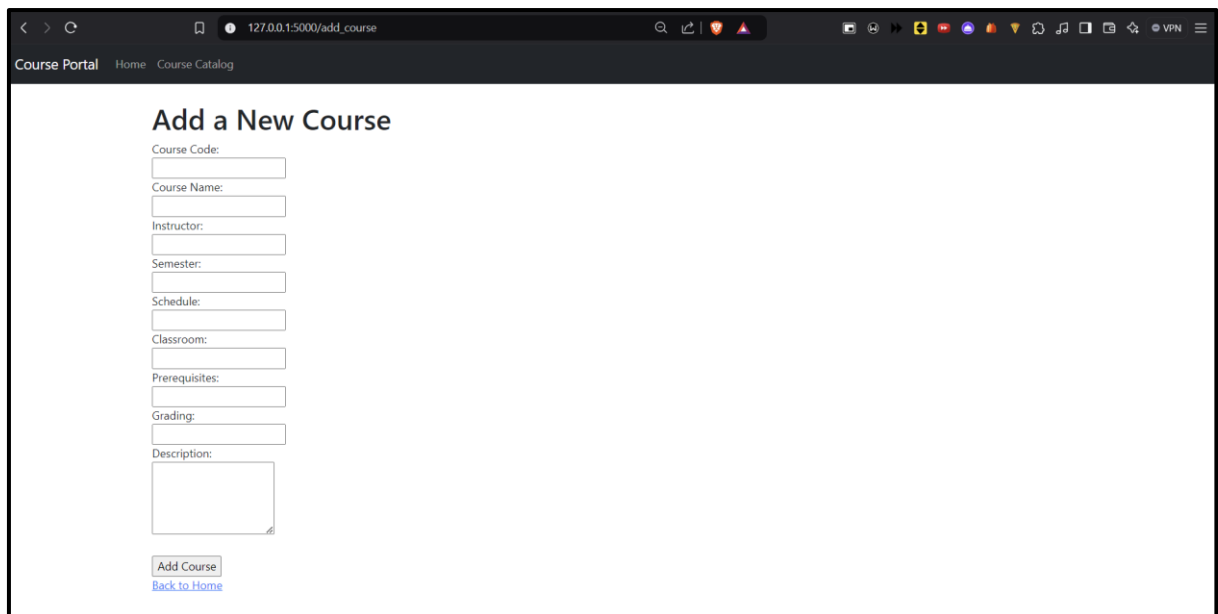
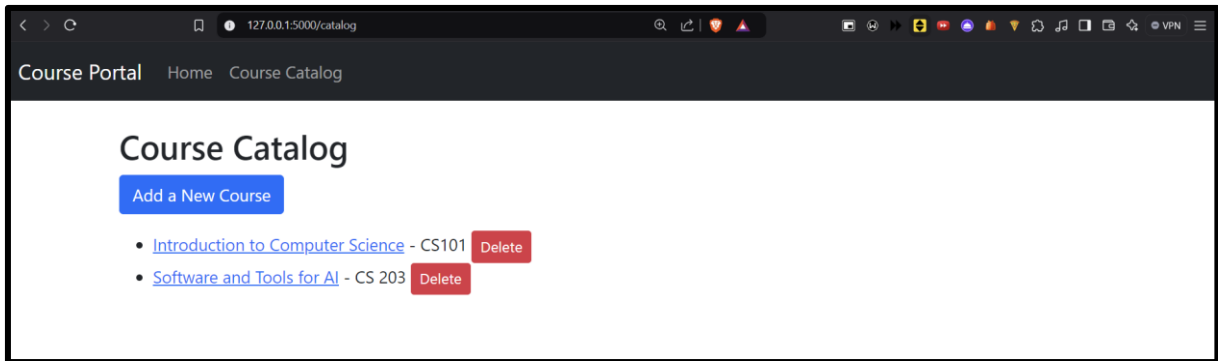


CS 203: Software Tools & Techniques for AI - Assignment 1

- Vansh Kumar (23110351)
- Aditya Jain (23110016)

GitHub Repository Link: <https://github.com/VanshOnGit/CS-203-Assignment-1>

1. Add Courses to the Catalog



Course Portal Home Course Catalog

127.0.0.1:5000/add_course

Course Code, Course Name and Instructor are required!

Add a New Course

Course Code:

Course Name:

Instructor:

Semester:

Schedule:

Classroom:

Prerequisites:

Grading:

Description:

[Back to Home](#)

When the Course Code/Course Name/Instructor is not entered, then the app gives an ERROR!

Error log message

```
app.py add_course.html app.log x README.md course_catalog.html
app.log
1 {"taskName": null, "message": "{ 'event': 'Form Validation Error', 'route': '/add_course', 'error': 'Missing required fields' }", "time": "2025-01-10T20:53:19.615795+00:00"}
2 |
```

Adding the course

Course Portal Home Course Catalog

127.0.0.1:5000/catalog

Course 'Digital Signal Processing' added successfully!

Course Catalog

- [Introduction to Computer Science](#) - CS101
- [Software and Tools for AI](#) - CS 203
- [Digital Signal Processing](#) - EE301

When all required details are entered, the new course is added.

We have also added a delete course button.

Course_add log

```
app.log
1
2 {"taskName": null, "message": "{ 'event': 'Course Added', 'course_code': 'EE301', 'course_name': 'Digital Signal Processing' }", "time": "2025-01-10T20:59:20.571104+00:00"}
3 {"taskName": null, "message": "{ 'event': 'Page Rendered', 'route': '/catalog', 'total_courses': 4, 'processing_time': 0.01156306266784668 }", "time": "2025-01-10T20:59:20.580461+00:00"}
4
```

OpenTelemetry Tracing, Exporting Data to Jaeger and Python Logging

render-course-catalog span is added to display operations in Course Catalog

```
@app.route('/catalog')
def course_catalog():
    start_time = time.time() # measure processing time
    requests_count['catalog'] += 1 # increment route requests count

    with tracer.start_as_current_span("render-course-catalog") as span:
        span.set_attribute("http.method", request.method) # log HTTP method
        span.set_attribute("http.url", request.url) # log request URL
        span.set_attribute("user.ip", request.remote_addr) # log user IP
        span.set_attribute("requests.count", requests_count['catalog']) # log request count
        span.add_event("Loading course catalog") # log event: catalog loading

    courses = load_courses() # load courses from JSON
    span.set_attribute("total_courses", len(courses)) # log number of courses
    span.set_attribute("processing_time", time.time() - start_time) # log processing time

    logger.info({
        "event": "Page Rendered", # page render event
        "route": "/catalog", # accessed route
        "total_courses": len(courses), # number of courses
        "processing_time": time.time() - start_time # time taken
    })

    return render_template('course_catalog.html', courses=courses) # render catalog page
```

render-course-catalog span Jaeger

The screenshot displays the Jaeger UI interface. At the top, navigation tabs include "JAEGER UI", "Search", "Compare", and "System Architecture". The main header shows a dropdown menu set to "course-catalog-service: GET /catalog" with a trace ID of "1479ec0". Below this, a summary bar indicates the trace started on January 11, 2025, at 02:09:21.304, with a duration of 8.91ms, one service, two depths, and two total spans.

The left sidebar lists services under "Service & Operation". The selected service is "course-catalog-service" with operation "GET /catalog". A sub-selection shows "course-catalog-service" with operation "render-course-catalog".

The right pane details the selected span, "render-course-catalog", which took 7.91ms. It includes a "Tags" section with various metadata such as http method (GET), url (/127.0.0.1:5000/catalog), and processing time (0). A "Process" section identifies it as belonging to the "course-catalog-service" using the "opentelemetry" SDK in Python version 1.29.0. The logs section contains one entry: "1ms: message = Loading course catalog".

course_catalog span Logging

```
app.py  add_course.html  app.log  index.html  course_catalog.html
app.log
1
2 {"taskName": null, "message": "{ 'event': 'Page Rendered', 'route': '/catalog', 'total_courses': 4, 'processing_time': 0.0}", "time": "2025-01-10T21:57:55.827451+00:00"}
3
```

view-course-details span is added to track the retrieval & rendering of course_details

```
@app.route('/course/<code>')
def course_details(code):
    start_time = time.time() # start timer
    with tracer.start_as_current_span("view-course-details") as span:
        span.set_attribute("http.method", request.method) # log HTTP method
        span.set_attribute("http.url", request.url) # log request URL
        span.set_attribute("user.ip", request.remote_addr) # log user IP
        span.set_attribute("course_code", code) # log course code

        courses = load_courses() # load courses from JSON
        course = next((course for course in courses if course['code'] == code), None) # find course

        if not course: # if course not found
            error_count['course_details'] += 1 # increment error count
            span.add_event("Course not found") # log missing course
            span.set_attribute("error_count", error_count['course_details']) # log error count

            logger.error({
                "event": "Course Not Found", # log event
                "route": "/course/<code>", # log route
                "course_code": code # log course code
            })

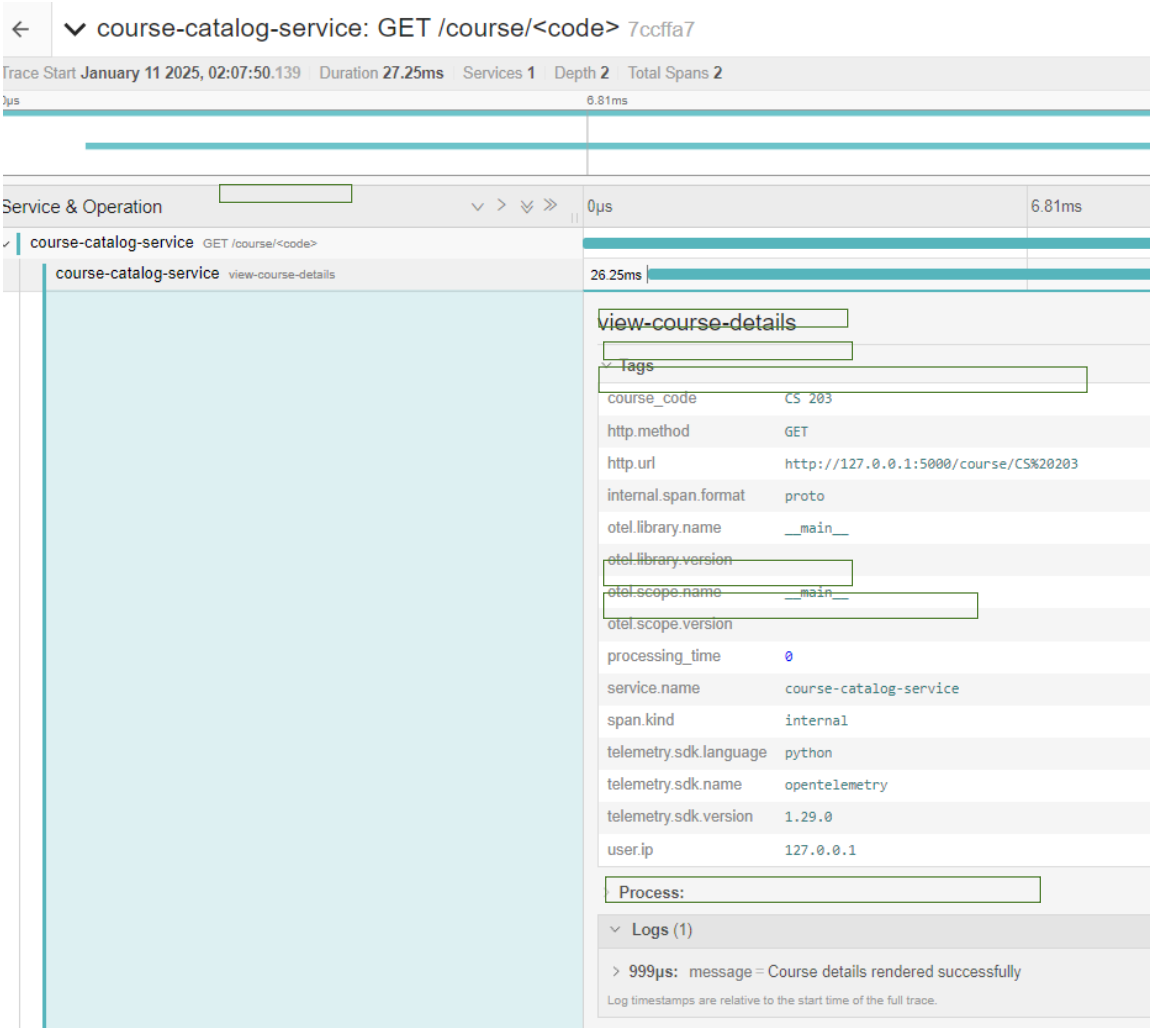
            flash(f"No course found with code '{code}'.", "error") # show error message
            return redirect(url_for('course_catalog')) # redirect to catalog

        span.add_event("Course details rendered successfully") # log render success
        span.set_attribute("processing_time", time.time() - start_time) # log time taken

        logger.info({
            "event": "Course Details Rendered", # log event
            "course_code": course['code'], # log course code
            "processing_time": time.time() - start_time # log processing time
        })

    return render_template('course_details.html', course=course) # render course details page
```

view-course-details span Jaeger



Viewing course details Logging

```
app.log
1
2 {"taskName": null, "message": "{\"event\": 'Course Details Rendered', 'course_code': 'CS101', 'processing_time': 0.00099945068359375}\", \"time\": \"2025-01-10T21:49:29.697039+00:00\"}"}
3
```

Please Move to Next Page

Please Move to Next Page

add-course span is added to trace the course addition process

```
@app.route('/add_course', methods=['GET', 'POST'])
def add_course():
    start_time = time.time() # start timer
    requests_count['add_course'] += 1 # count add requests
    with tracer.start_as_current_span("add-course") as span:
        span.set_attribute("http.method", request.method) # log HTTP method
        span.set_attribute("http.url", request.url) # log request URL
        span.set_attribute("user.ip", request.remote_addr) # log user IP
        span.set_attribute("requests.count", requests_count['add_course']) # log request count

    if request.method == 'POST': # check if form submitted
        course = {
            'code': request.form.get('code', '').strip(), # get course code
            'name': request.form.get('name', '').strip(), # get course name
            'instructor': request.form.get('instructor', '').strip(), # get instructor
            'semester': request.form.get('semester', '').strip(), # get semester
            'schedule': request.form.get('schedule', '').strip(), # get schedule
            'classroom': request.form.get('classroom', '').strip(), # get classroom
            'prerequisites': request.form.get('prerequisites', '').strip(), # get prerequisites
            'grading': request.form.get('grading', '').strip(), # get grading info
            'description': request.form.get('description', '').strip() # get description
        }

        if not course['code'] or not course['name'] or not course['instructor']: # check required fields
            error_count['add_course'] += 1 # count errors
            span.add_event("Validation failed: Missing required fields") # log validation error
            span.set_attribute("error_count", error_count['add_course']) # log error count

            logger.warning({
                "event": "Form Validation Error", # log form error
                "route": "/add_course", # log route
                "error": "Missing required fields" # log error reason
            })

            flash('Course Code, Course Name and Instructor are required!', 'error') # show error
            return render_template('add_course.html') # reload form

        save_courses(course) # save course data
        span.add_event("Course saved successfully") # log save success
        span.set_attribute("processing_time", time.time() - start_time) # log time taken
        logger.info({
            "event": "Course Added", # log add event
            "course_code": course['code'], # log course code
            "course_name": course['name'] # log course name
        })
        flash(f"Course '{course['name']}' added successfully!", "success") # show success message
        return redirect(url_for('course_catalog')) # go to catalog

    return render_template('add_course.html') # show add form
```

add-course span Jaeger

JAEGER UI

Search

Compare

System Architecture

<

▼ course-catalog-service: POST /add_course 15c392a

Trace Start January 11 2025, 02:07:21.106

Duration 19.88ms

Services 1

Depth 2

Total Spans 2

0µs

4.97ms

9

Service & Operation

▼ > ⌵ >>

0µs

4.97ms

▼ course-catalog-service POST /add_course

course-catalog-service add-course

add-course

▼ Tags

http.methodPOST

http.urlhttp://127.0.0.1:5000/add_course

internal.span.formatproto

otel.library.name__main__

otel.library.version

otel.scope.name__main__

otel.scope.version

processing_time0.019879817962646484

requests.count1

service.namecourse-catalog-service

span.kindinternal

telemetry.sdk.languagepython

telemetry.sdk.nameopentelemetry

telemetry.sdk.version1.29.0

user.ip127.0.0.1

> Process:

▼ Logs (1)

> 19.88ms: message = Course saved successfully

Log timestamps are relative to the start time of the full trace.

add-course Logging

app.py

add_course.html

app.log

README.md

course_catalog.html

app.log

1 {"taskName": null, "message": "{\n \"event\": \"Course Added\", \n \"course_code\": \"EE301\", \n \"course_name\": \"Digital Signal Processing\" \n}, \n \"time\": \"2025-01-10T20:59:20.571104+00:00\" \n}"

2 {"taskName": null, "message": "{\n \"event\": \"Page Rendered\", \n \"route\": \"/catalog\", \n \"total_courses\": 4, \n \"processing_time\": 0.01156306266784668 \n}, \n \"time\": \"2025-01-10T20:59:20.588461+00:00\" \n}"

3

Thank You
CS203 Assignment 1

Vansh Kumar (23110351)
Aditya Jain (23110016)