GenAI Conversation Export Logs

Module: COMP3011 – Web Services and Web Data
Student: Nathaniel Sebastian (sc232ns)
Date: 5th February 2026
Tools Used: Google Gemini (Antigravity), Claude (Anthropic), ChatGPT (OpenAI)

Summary

Tool	Purpose	Sessions
Google Gemini	Coding, debugging, test generation, security hardening	10
Claude	Documentation polish, refactoring review	2
ChatGPT	Early brainstorming	1

Test Count: 39 tests passing (verified via `pytest -q`)

Session 1: Architecture Planning

Tool: Google Gemini
Prompt: "Best architecture for FastAPI + SQLAlchemy REST API?"

AI Suggestion: Layered architecture (routes → CRUD → models → schemas).

Decision: Adopted. Created `app/api/routes.py` , `app/crud.py` , `app/models.py` , `app/schemas.py` .

Session 2: RSVP Storage (Creative Comparison)

Tool: Google Gemini
Prompt: "Embedded list vs separate table for RSVPs?"

AI Analysis:

- Embedded: Simpler, no JOINs, but no uniqueness constraint
- Separate: UNIQUE(event_id, attendee_id), timestamps, cascade delete

Decision: Separate table for data integrity.

Why Outstanding: Demonstrates understanding of relational constraints vs document patterns.

Session 3: Rate Limiting Strategy (Creative Comparison)

Tool: Google Gemini
Prompt: "In-memory vs Redis rate limiting for coursework?"

AI Analysis:

- In-memory: Simple, single-process, no infra
- Redis: Distributed, production-grade, requires setup

Decision: In-memory for coursework; documented Redis as future upgrade.

Trade-off Rationale: Appropriate for single-worker Render free tier.

Session 4: ETag Implementation (Research-Informed)

Tool: Google Gemini
Prompt: "How to implement ETag + If-None-Match correctly?"

AI Response: Referenced RFC 7232. Compute SHA256 of response body, wrap in quotes, check If-None-Match header, return 304 with no body.

Implementation:

- Middleware intercepts GET 200 responses
- Computes ETag, sets header
- Returns 304 if match
- 304 includes ETag + X-Request-ID + security headers

Why Outstanding: Demonstrates HTTP standards knowledge beyond basic CRUD.

Session 5: Error Sanitization

Tool: Google Gemini
Prompt: "How to prevent exception leakage in 500 responses?"

Solution: Remove `detail=str(e)` from handlers. Middleware catch-all returns generic message + request_id. Log exception server-side with `exc_info=True`.

Implementation: Admin import endpoint has no try/except; exceptions bubble to middleware.

Session 6: Middleware Ordering

Tool: Google Gemini
Prompt: "Security headers not applied to 429 responses?"

Problem: Early returns bypassed `add_headers` call.

Solution: Refactored to call `add_headers()` on ALL response paths (rate limit, exception, normal).

Failures & Corrections

AI Failure	Impact	My Fix
Missing `requests` in requirements.txt	ModuleNotFoundError	Added manually
Placeholder test with `pass`	False coverage	Rewrote with assertions
Deprecated `Query(regex=...)`	Warning	Changed to `pattern=...`
Headers not on 429	Missing security	Fixed middleware flow

Validation Steps

1. **Clean install test:** `rm -rf venv && python -m venv venv && pip install -r requirements.txt`
2. **Full test suite:** `pytest -q` → 39 passed
3. **Manual curl:** Register → Login → Create → RSVP → Analytics
4. **Admin RBAC:** Non-admin → 403; admin → 200
5. **ETag:** First GET → ETag; second with If-None-Match → 304
6. **Rate limit:** Flood → 429 with request_id

Conclusion

AI accelerated development ~3× but required manual verification. Critical bugs caught via clean-install testing.

COMP3011 CW1 Submission – 5th February 2026