School of Computer Science: Assessment Brief

Module title	Web Services and Web Data
Module code	COMP3011
Assignment title	Coursework 1: Individual Web Services API Development Project
Assignment type and description	Coursework – individual practical project with oral examination
Rationale	This coursework challenges you to design, implement, and present a fully functional data-driven web API with database integration. It assesses your ability to apply software engineering principles to real-world API design, justify design choices, demonstrate curiosity, creativity, and independence in software development, and present your work effectively in an oral examination.
Word limit and guidance	Assessment is based on a 10-minute oral examination (5 minutes presentation + 5 minutes Q&A) supported by submitted materials through GitHub repository, API documentation, and technical report (maximum 5 pages).
Use of GenAI in this assessment	**GREEN: AI has an integral role and should be used as part of the assessment.** You can use GenAI as a primary tool throughout the assessment process. You must declare all tools used, purposes, and attach examples of exported conversation logs as supplementary material. Using GenAI for debugging, planning or proof-reading is fully permitted. Higher grades will be awarded for creative, high-level use of GenAI (e.g., exploring alternatives, reimagining cutting-edge solutions). Non-declared use of GenAI constitutes academic misconduct.

Weighting	30% of overall module mark
Submission deadline	13 March 2026 (via Minerva). The maximum permitted extension due to mitigating circumstances is one week.
Submission method	Electronic submission through GitHub and Minerva
Feedback provision	Oral feedback with mark breakdown and rubric-based written comments via email within 2 weeks of the final presentation date
Learning outcomes assessed	• Application of software engineering principles to API design • Justification of design choices and stack selection • Demonstration of curiosity, creativity, and independence in software development • Effective presentation of technical work in an oral examination
Module lead	Dr Ammar Alsalka (M.A.Alsalka@leeds.ac.uk)
Other Staff contact	Mr Omar Choudhry (O.Choudhry@leeds.ac.uk)

1. **Assignment guidance**

   You will develop your API individually. The assessment encourages creativity, good design practice, and reflective learning in the era of Generative AI (GenAI). The advent of GenAI has unlocked frontiers once deemed unattainable. Instead of investing significant time and effort in mastering low-level programming concepts and techniques, computer science students and practitioners can now leverage AI to focus on building complex, creative, and advanced solutions.

   Therefore, this coursework is intentionally made open-ended to foster independent inquiry and innovation. While students can still ground their design and implementation in the foundational tools and technology stacks introduced in the module, higher grades will be awarded for solutions that demonstrate originality, advanced methodologies, and integration of contemporary technologies. Novel approaches that extend beyond the basic requirements, such as incorporating modern frameworks, architectural patterns, or emerging standards, will be recognised as evidence of critical engagement and technical sophistication.

   (a) Assessment Overview

   Your grade will be assessed through an **Oral Examination** supported by a comprehensive review of your submitted materials. The oral exam will last **10 minutes in total**, consisting of:

   - **5 minutes**: individual presentation and demonstration of your project.
   - **5 minutes**: structured questions from the examiners about your implementation, design decisions, and supporting materials.

   The coursework mark is composed of **75% for content**, **15% for quality of presentation**, and **10% for Q&A performance**. Examiners will review your GitHub repository (including commit history), API documentation, and technical report before, during, and after the oral examination. The final mark will be determined through examiner discretion, taking into account both your oral presentation performance and the quality of all submitted materials. You must be prepared to answer questions about any aspect of your submission.

(b) Oral Presentation Schedule

Oral presentations will take place during the week commencing **23rd March 2026** through to **Friday 27th March 2026**. Each student will be allocated a 10-minute presentation slot.

Important attendance requirements:

- **You must arrive 10 minutes before your scheduled start time and make sure all your devices, program code, and presentation material (e.g. slides, videos, etc.) are ready and working as expected.**
- **In case of emergency, you must send an email notification (except where additional considerations apply).**
- **Failure to notify by email in case of emergency will result in automatic failure of the oral examination component.**

(c) Deliverable Requirements

You must provide the following deliverables:

(i) Code Repository

- Public GitHub repository with visible commit history demonstrating consistent version control.
- Includes a clear `README.md` describing setup instructions and project overview.
- Code must be runnable and correspond to the version presented in the oral exam.
- Missing version control or `README.md` or non-runnable code will result in this component being graded as a Fail

(ii) API Documentation

- Generated or manually written documentation (e.g. Swagger UI, Postman, Markdown).
- Clearly describes all available endpoints, parameters, and response formats.

- Includes example requests and expected responses (JSON or equivalent[1]).
- Documents authentication process and error codes where applicable.
- Must be referenced in the `README.md` file as a PDF file (if generated as an HTML file, then convert to PDF).
- Missing API documentation will result in this component being graded as a Fail.

(iii) Technical Report

- Concise written report outlining key design and architectural choices. We want to see your personal choices for the technology stack.
- Justifies the decision of programming language, frameworks, and databases.
- Reflects on challenges, testing approach, and lessons learned.
- Includes limitations and potential areas for improvement or future development.
- Contains a Generative AI declaration and thoughtful analysis of its usage if used.
- Missing technical report or GenAI declaration will result in this component being graded as a Fail.

(iv) Presentation Slides

- PowerPoint presentation slides that will be used to present your work in the oral exam.
- Presentation slides must adhere to commonly known guidance on quality of good presentations. See this link for example: https://library.leeds.ac.uk/info/1401/academic-skills/130/presentations-oral

[1]You are permitted to use other formats provided you justify these in your supporting report document.

2. **Use of GenAI**

   This is a **Green Light Assessment**. You may use GenAI tools (e.g. Secure Microsoft Copilot) under the following rules:

   - You **must declare** all tools used, purposes, and attach some examples of exported conversation logs as supplementary material.
   - Using GenAI for debugging, planning or proof-reading is **fully permitted**.
   - Non-declared use of GenAI constitutes **academic misconduct**.

   The following table shows how different levels of GenAI usage relate to grade bands:

Grade Band	**GenAI Usage Level**
40–49 (Pass)	Generative AI is employed in an unsystematic and arbitrary manner.
50–59 (Satisfactory)	Low-level use of AI, e.g. to aid in writing or debugging code.
60–69 (Good)	Employs Generative AI in a methodologically sound manner.
70–79 (Very Good)	Medium level use of GenAI, e.g. to understand technology and new concepts.
80–89 (Excellent)	High level use of GenAI to aid creative thinking and solution exploration.
90–100 (Outstanding)	Creative application of Generative AI, such as exploring high-level alternatives and reimagining the design of cutting-edge solutions.

3. **Assessment tasks**

(a) Minimum Technical Requirements

To achieve a **Pass (i.e. 40+)**, your API must at a minimum:

- implement at least one data model involving all four elements of CRUD (Create, Read, Update and Delete) functionality linked to a database.[2] You can base your design on a RESTful (Representational State Transfer) API. However, you can also use a GraphQL-based API if you think this is more suitable for the problem, but you must explain the rationale for using GraphQL.
- includes at least four API endpoints accessible via HTTP requests
- handle user inputs and return appropriate JSON responses. You are permitted to use other formats, provided you justify these in your supporting report document.
- use correct status/error codes, according to industry convention
- be demonstrable via local execution, web hosting (e.g. PythonAnywhere), or a Model Context Protocol (MCP) server.

You can base your implementation on Django, but you can also use other suitable stacks (e.g. Python `FastAPI`, Node.js `Express`, Go `Fiber`). You must justify your technical choices and motivations.

(b) Example Project Ideas

- A book metadata and recommendation API integrating a public dataset (e.g. Goodreads or Google Books) and providing analytic endpoints such as genre trends, rating distributions, and personalised suggestions.
- An environmental and urban climate statistics API that stores historical weather, air quality, and temperature data, enabling users to query aggregated metrics, anomalies, and city-level trends.
- A productivity and habit analytics API that tracks tasks, habits, time logs, and completion patterns, offering endpoints for streak calculations, productivity heatmaps, and behavioural summaries.

[2]You are permitted to use any type of SQL database, but you should not use NoSQL databases except with clear reasoning in your supporting report document.

- A public transport reliability and delay analytics API using open UK transport datasets, providing CRUD for user-reported incidents and endpoints for route-level reliability scores, delay patterns, and temporal performance summaries.
- A nutrition and recipe analytics API backed by open food datasets (e.g. USDA or UK FSA), supporting ingredient/recipe CRUD and computing calories, macronutrients, allergen warnings, and cost or difficulty estimates.
- A housing market and rental insights API integrating ONS or Land Registry data, offering CRUD for listings and locations, plus analytics such as median rent, affordability indices, or regional price trends.
- A cultural events and tourism insights API using public datasets on museums, festivals, and heritage sites, enabling CRUD for events and providing popularity metrics, seasonal trends, and location-based recommendations.
- A sports performance and match statistics API backed by public sports datasets (e.g. football, cricket, or F1), supporting CRUD for teams, players, and matches, alongside analytical endpoints for leaderboards, performance summaries, and win probabilities.

(c) Data Sources

You are encouraged to use publicly available datasets for your project. Below are some suggested platforms where you can access suitable data:

- **Kaggle** (https://www.kaggle.com/datasets) – Large collection of datasets across various domains, including books, weather, health, and more.
- **data.gov.uk** (https://www.data.gov.uk/) – UK government open data portal with datasets on transport, health, education, and public services.
- **OpenWeatherMap** (https://openweathermap.org/api) – Weather data API that can be used to collect historical and current weather information.
- **Google Dataset Search** (https://datasetsearch.research.google.com/) – Search engine for publicly available datasets.

- **UCI Machine Learning Repository** (https://archive.ics.uci.edu/) – Datasets commonly used for machine learning and data analysis projects.
- **NHS Digital** (https://digital.nhs.uk/data) – Health and care data from the UK National Health Service.

Ensure that any dataset you use is appropriately licensed for academic use, and clearly cite the source in your technical report.

(d) Using AI to Work with Datasets

You are encouraged to leverage Generative AI tools to help you discover, process, and import datasets into your database. A recommended workflow includes:

(i) Dataset Discovery: Use AI to search online for datasets in areas you are passionate about or interested in exploring. Ask AI to suggest relevant datasets, data sources, or domain-specific repositories.

(ii) Download and Explore: Download an interesting dataset and examine its structure, format, and contents to understand what data is available.

(iii) Generate Import Scripts: Provide a subset of your dataset to AI and ask it to help you create a script to import and store the data in your SQL database. This can significantly accelerate the data preparation process.

This approach encourages creative exploration while demonstrating effective use of AI as a productivity tool. Remember to declare your use of AI tools in your technical report, as outlined in Section 2.

4. **General guidance and study support**

   Detailed information and guidance are available in the module's learning resources on Minerva. If you encounter difficulties or have questions about the assignment:

   - Attend module practical lab sessions where teaching staff can provide guidance;
   - Post questions on the module discussion forum on Minerva;
   - Contact the module leader or teaching staff during office hours.

5. **Assessment criteria and marking process**

Band	Content (75%)	Presentation (15%)	Q&A (10%)
40–49 (Pass)	• Working CRUD operations with the database • Basic API structure implemented • Minimal documentation provided • Limited design justification • server-side code not deployed on an external platform. • Commit history visible but limited • Generative AI is employed in an unsystematic and arbitrary manner.	• Presentation delivered but lacks structure • Minimal use of visuals • Reading from slides • Poor time management	• Struggles to answer basic questions • Limited understanding of implementation • Cannot explain design choices

Continued on next page

Band	Content (75%)	Presentation (15%)	Q&A (10%)
50–59 (Satisfactory)	• Complete API with documentation • Basic authentication present • Demonstrates understanding of architecture • Clear technical report • Regular commit history • Hosted on an external web server, e.g. PythonAnywhere	• Coherent presentation structure • Some visual aids used • Mostly descriptive content • Adequate time management • Low-level use of AI, e.g. to aid in writing or debugging code.	• Answers basic questions adequately • Can explain main features • Limited depth in responses
60–69 (Good)	• Well-documented API with authentication • Effective error handling • Clear stack choice justification • Evidence of testing approach • Consistent version control • Employs Generative AI in a methodologically sound manner.	• Well-structured presentation • Good use of diagrams/screenshots • Clear verbal delivery • Covers all deliverables	• Answers questions confidently • Explains design decisions clearly • Shows good understanding of code

Continued on next page

Band	Content (75%)	Presentation (15%)	Q&A (10%)
70–79 (Very Good)	• Clean, modular code design • advanced features, e.g. MCP-compatible. • Comprehensive documentation • Strong version-control discipline • Thorough testing demonstrated • Professional deployment • medium level use of GenAI, e.g. to understand technology and new concepts.	• Professional presentation • Effective use of visuals/demos • Engaging delivery style • Excellent timing	• Provides detailed, technical answers • Discusses trade-offs and alternatives • Shows deep code understanding

Continued on next page

Band	Content (75%)	Presentation (15%)	Q&A (10%)
80–89 (Excellent)	• Exemplary code quality and architecture • Advanced security implementation • Comprehensive testing suite • Creative data design • Excellent documentation • high level use of GenAI to aid creative thinking and solution exploration.	• Highly professional delivery • Compelling visual aids • Confident and articulate • Demonstrates expertise	• Provides insightful responses • Discusses advanced concepts • Demonstrates independent thinking

Continued on next page

Band	Content (75%)	Presentation (15%)	Q&A (10%)
90–100 (Outstanding)	• Exceptional originality and innovation • Novel data integration or features • Publication-quality documentation • Demonstrates genuine research curiosity • Creative application of Generative AI, such as exploring high-level alternatives and reimagining the design of cutting-edge solutions	• Exceptional presentation quality • Outstanding visual communication • Inspirational delivery • Professional-grade polish	• Provides expert-level analysis • Discusses cutting-edge approaches • Shows exceptional mastery

Important Notes:

- Presentations that are overly text-heavy or read verbatim from slides will reduce your presentation skills score.
- Students are expected to use visual elements such as screenshots, diagrams, or recorded demos.
- Your PowerPoint presentation should include dedicated sections covering: version control practices, API documentation, technical report highlights, and all deliverables.
- During the examination, examiners will navigate through your GitHub repository, review your commit history, examine your API documentation, and assess your technical report while asking questions.
- After your oral presentation, you will receive detailed feedback via email. We aim to provide feedback by email within 2 weeks of the fi-

nal presentation date. This timeframe allows examiners to thoroughly review all submitted materials, apply appropriate discretion in marking, and ensure that the final grade accurately reflects both your oral presentation performance and the quality of your complete submission.

6. **Presentation and referencing**

   Your report and oral presentation must:

   - Use clear English and professional communication.
   - Be logically structured with labelled sections.
   - Include diagrams where appropriate.
   - Reference any libraries, tutorials, or datasets used.

   The quality of written English will be assessed in this work. As a minimum, you must ensure:

   - Paragraphs are used
   - There are links between and within paragraphs although these may be ineffective at times
   - There are (at least) attempts at referencing
   - Word choice and grammar do not seriously undermine the meaning and comprehensibility of the argument
   - Word choice and grammar are generally appropriate to an academic text

   These are pass/fail criteria. So irrespective of marks awarded elsewhere, if you do not meet these criteria you will fail overall.

7. **Submission requirements**

(a) What to Submit via Minerva

- **Technical Report (PDF)** containing:
  - Link to your public GitHub repository
  - Link to your API documentation (accessible through the GitHub repository)
  - Link to your presentation slides, and other visuals (hosted on any platform, e.g., Google Drive, OneDrive, etc.)
  - All required sections as outlined in the Pass/Fail Components
  - GenAI declaration, analysis, and conversation logs as an appendix
- **Presentation Slides (PPTX)** including dedicated sections on:
  - Version control practices and commit history
  - API documentation overview
  - Technical report highlights
  - All deliverables (code repository, API docs, technical report)

(b) GitHub Repository Requirements

Your GitHub repository must contain:

- Versioned source code with clear commit history
- `README.md` file with setup instructions and project overview
- API documentation PDF
- All code must be runnable and correspond to the version presented in the oral exam

(c) Late Submissions

- Late submissions without approved mitigation incur a **5% penalty per day**.
- **Maximum extension period for mitigating circumstances is 1 week.**

8. **Academic misconduct and plagiarism**

   All work must comply with the University of Leeds standards on academic integrity. You are expected to distinguish your own work from others' and correctly acknowledge the use of AI tools.

   - Leeds students are part of an academic community that shares ideas and develops new ones.
   - You need to learn how to work with others, how to interpret and present other people's ideas, and how to produce your own independent academic work. It is essential that you can distinguish between other people's work and your own, and correctly acknowledge other people's work.
   - All students new to the University are expected to complete an online Academic Integrity tutorial and test, and all Leeds students should ensure that they are aware of the principles of Academic integrity.
   - When you submit work for assessment it is expected that it will meet the University's academic integrity standards.
   - If you do not understand what these standards are, or how they apply to your work, then please ask the module teaching staff for further guidance.

   By submitting this assignment you are confirming that the work is a true expression of your own work and ideas and that you have given credit to others where their work has contributed to yours.

9. **Assessment/marking criteria grid**

Assessment Criteria	**Maximum Mark**	**Mark Achieved**	**Feedback**
Content (75%)			
API Functionality & Implementation	25		
Code Quality & Architecture	20		
Documentation (API & Technical Report)	12		
Version Control & Deployment	6		
Testing & Error Handling	6		
Creativity & Level of GenAI usage	6		
Presentation Skills (15%)			
Structure & Clarity	5		
Visual Aids & Delivery	5		
Time Management	5		
Q&A Performance (10%)			
Depth of Understanding	4		
Ability to Explain Design Decisions	3		
Response to Technical Questions	3		
Total Mark	**100**		