Innovation in Cloud Engineering

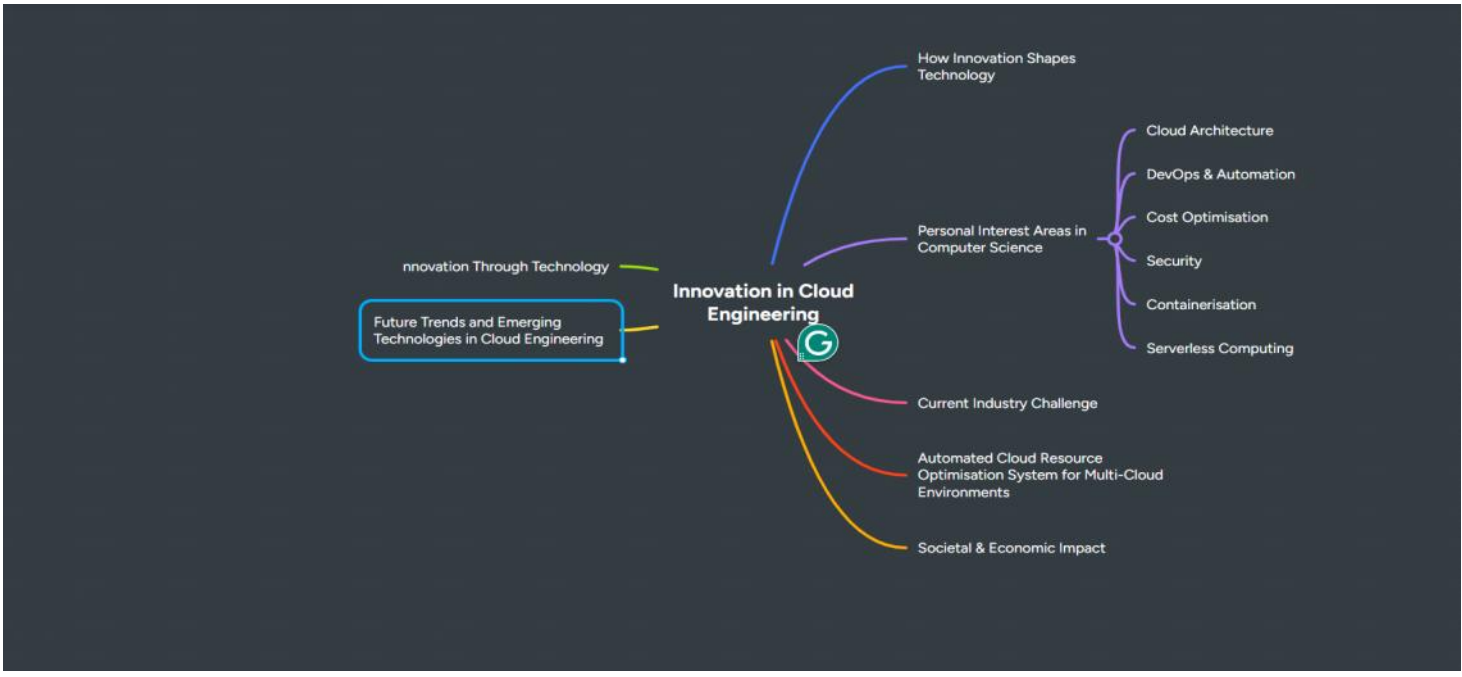
13 November 2025 13:43

Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Student name: Andy Onyedikachukwu Ibeh
Student ID: 2335550
Programme: HN/BSc Computing portfolio
Module: Changemakers: Creativity and Value Creation
Module Code: ACCA5033
Assignment Title: Assignment 1: Innovation & Research.
Lecturer: Dr.Fahad Ahmad
Submission deadline: Friday, 28th November 2025

TASK 1: IDEATION & MIND MAP

13 November 2025 13:47



Branch 1: How Innovation Shapes Technology

Innovation fundamentally transforms how society interacts with technology and drives economic growth through technological advancement. Cloud computing represents one of the most significant innovations of the 21st century, revolutionising how organisations store, process, and access data [1]. The evolution from traditional on-premises infrastructure to cloud-based solutions demonstrates how creative problem-solving addresses real world challenges of scalability, accessibility, and cost management. This transformation has enabled small businesses to access enterprise-level computing resources, democratising technology access across different economic sectors [2].

The role of creativity in cloud engineering extends beyond technical implementation to encompass business model innovation. Companies like Amazon Web Services, Microsoft Azure, and Google Cloud Platform pioneered the Infrastructure as a Service (IaaS) model, fundamentally changing how organisations budget for and consume IT resources. This shift from capital expenditure to operational expenditure models represents creative thinking applied to business operations, enabling organisations to scale resources dynamically based on demand rather than maintaining fixed infrastructure investments [3].

Branch 2: Personal Interest Areas in Computer Science

Cloud Architecture: Multi-cloud and hybrid cloud environments

DevOps & Automation: CI/CD pipelines, Infrastructure as Code

Cost Optimisation: Resource monitoring and automated scaling

Security: Cloud security frameworks and compliance

Containerization: Docker, Kubernetes orchestration

Serverless Computing: Function-as-a-Service implementations

Branch 3: Current Industry Challenges

Cloud cost management has emerged as a critical concern, with organisations reporting that 30-35% of cloud spending is wasted on unused or underutilised resources [4]. The complexity of multi-cloud environments, where organisations utilise services from multiple providers simultaneously, creates significant challenges in monitoring, optimisation, and governance. These challenges present opportunities for innovative solutions that leverage automation and machine learning to optimise resource allocation dynamically.

Branch 4: Automated Cloud Resource Optimisation System for Multi-Cloud Environments

This project addresses the pressing need for intelligent resource management across multiple cloud platforms. The system will analyse usage patterns, predict demand fluctuations, and automatically adjust resource allocation to minimise costs while maintaining performance standards. Key innovation aspects include machine learning algorithms for usage prediction, automated policy enforcement, and real-time cost analysis dashboards. The project combines technical skills in cloud APIs, data analytics, and software development while addressing a genuine industry pain point that affects organisations globally.

Branch 5: Societal & Economic Impact

Effective cloud resource optimisation contributes to environmental sustainability by reducing unnecessary computing resource consumption, thereby lowering energy consumption and carbon footprints of data centres. Economically, organisations can redirect saved resources toward innovation and growth rather than infrastructure overhead. This aligns with broader societal goals of sustainable technology development and efficient resource utilisation [5].

Branch 6: Innovation Through Technology

The convergence of artificial intelligence, cloud computing, and automation creates opportunities for self-optimising systems that require minimal human intervention. This project embodies the principle that innovation emerges from identifying inefficiencies in current systems and developing creative solutions that leverage emerging technologies to address them.

Branch 7: Future Trends and Emerging Technologies in Cloud Engineering

The cloud computing ecosystem is evolving at a fast pace, propelled by emerging technologies that promise to reshape how organisations architect and consume cloud resources. Edge computing is gaining prominence as Internet of Things (IoT) devices proliferate, requiring data processing closer to the source to reduce latency and bandwidth costs. This shift from centralised cloud data centres to distributed edge nodes represents a fundamental architectural transformation that cloud optimisation systems must accommodate.

References

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, vol. 53, no. 6, pp. 50-50, 2011.

[2] M. Armbrust et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50-58, Apr. 2010.

[3] L. Wang et al., "Cloud computing: a perspective study," New Generation Computing, vol. 28, no. 2, pp. 137-146, 2010.

[4] Flexera, "State of the Cloud Report 2023," Flexera Software LLC, Tech. Rep., 2023.

[5] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in Proc. 10th IEEE/ACM Int. Conf. Cluster, Cloud and Grid Computing, 2010, pp. 826-831.

TASK 2: AIM & OBJECTIVES AND RESEARCH AREA

13 November 2025 13:49

Project Aim

To design, develop, and evaluate an automated cloud resource optimisation system that reduces operational costs by at least 25% in multi-cloud environments while maintaining service level agreement (SLA) compliance and system performance.

Project Objectives

To conduct a comprehensive literature review of existing cloud resource optimisation techniques, cost management strategies, and machine learning approaches for predictive resource allocation in multi-cloud environments.

To analyse current cloud resource utilisation patterns across major cloud platforms (AWS, Azure, GCP) and identify common inefficiencies and optimisation opportunities through case study analysis.

To design an architectural framework for an automated optimisation system that integrates with multiple cloud provider APIs and implements real-time monitoring, analysis, and resource adjustment capabilities.

To develop a prototype system incorporating machine learning algorithms for usage prediction, automated scaling policies, and cost-benefit analysis functionality using appropriate programming languages and cloud SDKs.

To evaluate the system's effectiveness through controlled testing environments, measuring cost reduction, response time impact, resource utilisation improvements, and SLA compliance rates.

To produce comprehensive documentation, including system architecture, implementation guidelines, user manuals, and recommendations for deployment in production environments.

Research Philosophy

This project will adopt a pragmatic research philosophy, drawing on elements of positivism. The sensible approach is most appropriate for this applied computing project because it focuses on practical problem-solving and real-world application rather than purely theoretical knowledge [1]. Pragmatism enables the integration of both quantitative data (cost metrics, performance statistics, and resource utilisation percentages) and qualitative insights (user feedback, system usability assessments) to comprehensively evaluate the solution's effectiveness [2].

The positivist elements come into play when measuring concrete, observable outcomes like expenditure decrease proportions, system response times, and resource allocation efficiency. These quantifiable metrics can be objectively measured and analysed to determine the system's success in achieving its stated objectives. This philosophical stance aligns with the engineering nature of the project, where empirical testing and measurable outcomes validate the proposed solution [3].

Search Tools and Research Databases

Academic Databases

IEEE Xplore Digital Library - secondary source for computer science and engineering research papers on cloud computing, optimisation algorithms, and distributed systems

ACM Digital Library - Essential for software engineering methodologies and computing research

SpringerLink - Access to computer science journals and conference proceedings

Google Scholar - Broad academic search across multiple publishers and preprint servers

Industry Resources

Cloud Provider Documentation - AWS Whitepapers, Azure Architecture Centre, Google Cloud Documentation

Gartner Research - Industry analysis and cloud market trends

GitHub - Open-source cloud optimization tools and implementation examples

Search Strategies

Secondary Search Terms:

cloud resource optimisation AND cost management

multi-cloud" AND automated scaling

machine learning AND cloud resource prediction

cloud cost optimisation AND algorithm

Boolean Operators Strategy:

Using AND to combine concepts (e.g., "cloud computing" AND "optimisation"), OR to expand searches (e.g., "AWS" OR "Azure" OR "GCP"), and NOT to exclude irrelevant results (e.g., "cloud" NOT "weather").

Date Filtering:

Prioritising research from 2019-2024 to ensure relevance to current cloud technologies and practices, while including seminal papers from earlier periods that established foundational concepts [4].

Citation Tracking:

Identifying highly-cited papers on cloud optimisation and reviewing their references to discover additional relevant research (backward citation tracking) and examining papers that cite them (forward citation tracking) to find the most recent developments in the field [5].

References

[1] J. W. Creswell and V. L. Plano Clark, Designing and Conducting Mixed Methods Research, 3rd ed. Thousand Oaks, CA: SAGE Publications, 2017.

[2] C. Robson and K. McCartan, Real World Research, 4th ed. Chichester: John Wiley & Sons, 2016.

[3] R. K. Yin, Case Study Research and Applications: Design and Methods, 6th ed. Thousand Oaks, CA: SAGE Publications, 2018.

[4] H. Zhang et al., "Guidelines for conducting systematic literature reviews in software engineering," Technical Report EBSE-2007-01, 2007.

[5] J. Webster and R. T. Watson, "Analysing the past to prepare for the future: Writing a literature review," MIS Quarterly, vol. 26, no. 2, pp. xiii-xxiii, 2002.

TASK 3: RESEARCH STRATEGIES

13 November 2025 13:51

Validity, Reliability, and Generalisability Strategies

Validity Enhancement:

To ensure this research produces valid results that accurately measure what it intends to measure, this project will employ multiple validation strategies. Construct validity will be established by clearly defining metrics such as "cost optimisation" (measured as percentage reduction in monthly cloud expenditure) and "system performance" (measured through response time, throughput, and availability metrics) [1]. Internal validity will be strengthened through controlled experimental designs where the optimisation system is tested against baseline scenarios with identical workloads, eliminating confounding variables that might influence results..

Reliability Enhancement:

Reliability ensures consistent results when measurements are repeated under similar conditions. The project will implement automated data collection processes to eliminate human measurement error and bias. All tests will be repeated multiple times (minimum three iterations) with results averaged to account for variability. Clear operational definitions and standardised testing procedures will be documented in detail, enabling future researchers to replicate the methodology exactly. Version control systems (Git) will track all code changes, ensuring reproducibility of the exact system configuration used during testing [3].

Ethical Considerations:

This project will adhere to ethical research practices, including data privacy, transparency, and responsible disclosure. If real organisational data is used for testing (anonymised), appropriate permissions and data protection measures will be implemented in compliance with GDPR and institutional ethics guidelines. Any limitations or potential negative impacts of automated resource management (such as potential service disruptions during optimisation) will be transparently documented. This research will consider the environmental impact of cloud computing and position cost optimisation as contributing to sustainability goals [4].

Research Strategies and Data Collection Methods

Mixed Methods Approach:

This project employs a mixed-methods research design combining quantitative performance measurements with qualitative assessment of system usability and practical applicability [5].

Quantitative Data Collection:

Experimental Testing

Controlled A/B testing comparing optimised vs. Non optimised environments

Load testing with varying workload intensities

Duration: 30-day testing periods to capture usage pattern variations

Sample size: Minimum 1000 resource allocation decisions per test scenario

Qualitative Data Collection:

Expert Interviews

Semi structured interviews with 5-7 cloud engineers and DevOps professionals

Topics: Current optimisation challenges, solution requirements, usability assessment

Duration: 45-60 minutes per interview

Pilot Data Collection Method

Pilot Study Design: A two-week pilot study will be conducted using a small scale cloud environment to test data collection instruments and refine the methodology before full implementation.

Pilot Environment:

Single cloud provider (AWS) with 3-5 compute instances

Simulated web application workload with variable traffic patterns

Budget: $200-300 for cloud resources

Pilot Data Collection:

Baseline measurement (Week 1): Monitor without optimisation

Optimised measurement (Week 2): Enable automated optimisation

Metrics collected: Hourly cost data, CPU/memory utilization, response times

Pilot Study Design:

A two-week pilot study will be conducted using a small scale cloud environment to test data collection instruments and refine the methodology before full implementation.

Pilot Results and Refinements (see Appendix A for detailed feedback):

Feedback indicated the need to include database resource optimisation (not just compute instances) and to extend monitoring intervals to capture weekly usage patterns. The full study will incorporate these refinements.

Quantitative and Qualitative Analysis Methods

Quantitative Analysis Techniques:

Descriptive Statistics

- Mean, median, standard deviation for cost and performance metrics
- Frequency distributions of resource utilisation patterns
- Time series visualisation of cost trends.

Performance Metrics Calculation

- Baseline Cost minus Optimised Cost)divided by Baseline Cost, multiplied by100
- Material utilisation Performance optimisation: (Active Operational Time / Total Provisioned Time) × 100.
- SLA compliance rate: (Time Within SLA / Total Time) × 100.

Qualitative Analysis Techniques:

Thematic Analysis

- Interview transcripts coded using NVivo or similar qualitative analysis software.
- Identification of recurring themes regarding optimisation challenges and requirements.
- Pattern recognition across different participant experiences.

Content Analysis

- Systematic categorisation of user feedback responses,
- Frequency analysis of mentioned usability issues or feature requests.
- Sentiment analysis of open-ended survey responses.

Framework Analysis

- Structured analysis matrix comparing findings against existing cloud optimisation frameworks.
- Gap analysis identifying areas where current solutions fall short.

References

[1] A. Bryman, Social Research Methods, 5th ed. Oxford: Oxford University Press, 2016.

[2] D. Silverman, Doing Qualitative Research, 5th ed. London: SAGE Publications, 2017.

[3] J. Oates, Researching Information Systems and Computing. London: SAGE Publications, 2006.

[4] British Computer Society, "Code of Conduct for BCS Members," 2022. [Online]. Available: https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/

[5] V. Braun and V. Clarke, "Using thematic analysis in psychology," Qualitative Research in Psychology, vol. 3, no. 2, pp. 77-101, 2006.

APPENDIX A: PILOT STUDY FEEDBACK

13 November 2025 14:50

Feedback from Cloud Engineering Practitioner 1

Background: Senior DevOps Engineer, 8 years' experience with AWS and Azure

Feedback on Methodology:

The two-week pilot duration is good for initial testing, but you should consider extending the full study to at least 4 weeks to capture monthly billing cycles and workload variations.

Include database instances (RDS/Azure SQL) in addition to compute instances – they're often the biggest cost drivers.

Consider testing during different times (business hours vs. off-peak) as optimisation opportunities vary significantly.

Suggested Metrics:

Add "time-to-optimisation" metric (how quickly the system responds to workload changes)

Include "false positive rate" for unnecessary scaling actions

Track "optimisation opportunity identification rate"

Overall Assessment: "The methodology is sound, but expand the scope to include more resource types and longer testing periods."

Feedback from Cloud Engineering Practitioner 2

Background: Cloud Solutions Architect, 6 years' experience with multi-cloud environments

Feedback on Data Collection:

5-minute aggregation intervals are appropriate for compute resources, but may be too granular for cost analysis – hourly aggregation might be more meaningful.

Ensure you're capturing data on spot/reserved instances separately from on-demand, as optimisation strategies differ significantly.

Include network egress costs in your cost tracking – often overlooked but can be substantial.

Suggested Improvements:

Implement anomaly detection to flag unusual usage patterns that might skew results

Add user-configurable cost thresholds and risk tolerance parameters

Consider including compliance and regulatory constraints in the optimisation logic

Overall Assessment:

Strong foundation, but needs more sophisticated cost breakdown and risk management considerations.

Refinements Incorporated from Pilot Feedback

Based on feedback received, the following refinements will be implemented in the full study:

Extended Duration: Full study period increased from 2 weeks to 4-6 weeks to capture complete billing cycles and diverse workload patterns.

Expanded Resource Scope: Include databases, storage services, and network resources in addition to compute instances.

Enhanced Metrics:

- Added time-to-optimisation metric.
- Added false positive rate tracking.
- Separated analysis for different instance purchasing options (on-demand, reserved, spot.
- Included network egress cost tracking.

Refined Data Aggregation:

- Maintain 5-minute intervals for performance monitoring.
- Use hourly aggregation for cost analysis.
- Implement daily and weekly summary reports.

Risk Management Features:

- User-configurable cost/performance trade-off parameters.
- Anomaly detection with alert mechanisms.
- Compliance constraint integration.

APPENDIX B: SUPPLEMENTARY INFORMATION

13 November 2025 14:51

Project Timeline

Phase 1: Research & Planning (Weeks 1-3)

- Literature review
- Requirement analysis

1. Architecture design

Phase 2: Development (Weeks 4-8)

- Core system development
- API integration
- ML model training

Phase 3: Testing & Evaluation (Weeks 9-11)

- Pilot testing
- Full-scale evaluation
- Performance analysis

Phase 4: Documentation & Finalization (Weeks 12-13)

- Technical documentation
- User guide creation
- Final report writing

Technology Stack

Development:

- Programming Languages: Python 3.9+, JavaScript
- Cloud SDKs: Boto3 (AWS), Azure SDK, Google Cloud Client Library
- Machine Learning: TensorFlow, Scikit-learn
- Data Analysis: Pandas, NumPy
- Visualisation: Matplotlib, Grafana

Infrastructure:

- Container Platform: Docker.
- Orchestration: Kubernetes (optional)
- CI/CD: GitHub Actions.
- Monitoring: Prometheus, CloudWatch.

Risk Assessment

Technical Risks:

- API rate limiting from cloud providers.
- Mitigation: Implement caching and request throttling.

Project Risks:

- Cloud service costs exceeding budget.
- Mitigation: Set strict spending alerts and use free-tier resources where possible.

Timeline Risks:

- Development delays due to complexity.
- Mitigation: Prioritise core features, use agile methodology with weekly sprints.