Final report

Carbon Calculator External API

Project code: L.ADP.2506

Prepared by: Norbert Feron
Cognizant

Date published: 20th December 2024

PUBLISHED BY
Meat and Livestock Australia Limited
PO Box 1961
NORTH SYDNEY NSW 2059

Meat & Livestock Australia acknowledges the matching funds provided by the Australian Government to support the research and development detailed in this publication.

Abstract

This project developed an External API service for MLA's Carbon Calculator to enable automated carbon emissions calculations by third-party systems. While the existing web-based calculator successfully digitized the Greenhouse Accounting Framework, it required manual data entry, limiting its integration potential.

The solution implemented a serverless API using AWS Lambda and Node.js, maintaining compatibility with the existing calculation engine while enabling programmatic access. The API supports all industry frameworks, methodology versioning, and comprehensive emissions reporting.

Key outcomes include deployment of a scalable, cost-effective service that processes requests within milliseconds while maintaining calculation accuracy. The solution enables seamless integration with external systems through documented OpenAPI specifications.

This enhancement empowers the industry with automated carbon accounting capabilities, supporting integration with farm management systems and enabling efficient large-scale emissions reporting across the sector.

Executive summary

Background

The MLA Carbon Calculator needed to expand its accessibility beyond manual web interface inputs to support automated carbon calculations via third-party applications. The primary question addressed was how to provide programmatic access to the calculator while maintaining calculation accuracy and methodology consistency. The target audience includes agricultural software providers, farm management systems, and organizations requiring automated emissions reporting.

Objectives

- Create an External API service that exposes the carbon calculation engine through RESTful endpoints
- Enable programmatic access while maintaining calculation accuracy and methodology versioning
- Implement secure authentication and usage monitoring
- Develop comprehensive API documentation for third-party integration

Methodology

The project implemented a serverless architecture using AWS Lambda and API Gateway as its foundation. A Node.js-based API service was developed to maintain compatibility with the existing CortexJS calculation engine, ensuring consistent results. The solution includes comprehensive OpenAPI specification documentation to support third-party developers, while security is managed through API key authentication and usage quotas. This approach enabled us to create a scalable, secure service that seamlessly integrates with existing systems.

Results/key findings

The project successfully delivered a scalable API service that processes requests within milliseconds, maintaining full calculation accuracy with the web calculator. The service operates independently of the existing web platform, ensuring zero interference with current operations, while keeping infrastructure costs aligned through serverless architecture.

Benefits to industry

The API enables agricultural businesses to automate their carbon accounting processes and integrate emissions calculations into their existing systems. This automation reduces manual effort, improves data accuracy, and facilitates large-scale emissions reporting across the sector.

Future research and recommendations

Two key areas require attention: upgrading the original Lambda ASP.NET Core API from the deprecated .NET 6 version before March 2025 and addressing the version disparity between current MLA Calculator methodologies and the latest PICCC GAF tool versions. It's recommended to align methodology versions with PICCC.org.au to ensure consistency across the industry.

Table of Contents

Abstract 2

Executive summary *3*

1. Background *5*

Current System Limitations **5**

Target Audience and Use Cases **5**

2. Objectives *5*

3. Methodology *5*

Architecture Design **5**

4. Results *6*

Technical outcomes **6**

Integration Success **6**

5. Conclusion *6*

Key findings **6**

Benefits to industry **6**

6. Future research and recommendations *6*

7. References *7*

8. Appendix *7*

Sprint review **7**

Swagger documentation **7**

1. Background

Current System Limitations

The MLA Carbon Calculator successfully digitized the Greenhouse Accounting Framework (GAF), converting traditional Excel-based calculations into a web-based tool. While effective for individual users, the system's reliance on manual data entry through a web interface created significant limitations for organizations needing to perform calculations at scale or integrate carbon accounting into their existing workflows.

The agricultural sector increasingly requires automated solutions for environmental reporting and carbon accounting. Organizations managing multiple properties or providing farm management software need programmatic access to perform calculations automatically, ensure consistency in methodology application, and integrate carbon metrics into their existing systems.

Target Audience and Use Cases

The primary target audience includes agricultural software providers, farm management systems, and organizations requiring automated emissions reporting. These stakeholders need to:

1. Integrate carbon calculations directly into their applications
2. Automate emissions reporting processes
3. Access calculations using specific methodology versions for compliance
4. Process large volumes of calculations efficiently

2. Objectives

The project set out to create an External API service that would expose the carbon calculation engine through RESTful endpoints while enabling version-controlled access to calculation methodologies. We aimed to implement secure authentication and usage monitoring systems while maintaining calculation accuracy and consistency with the web-based tool. A crucial objective was the development of comprehensive API documentation to facilitate third-party integration. All these objectives were successfully achieved through the project's implementation.

3. Methodology

Architecture Design

The solution implemented a serverless architecture utilizing AWS Lambda and API Gateway. The core architecture consists of a Node.js-based API service that maintains compatibility with the existing CortexJS engine. We implemented API Gateway to handle requests and security management, while serverless Lambda functions ensure scalable computation. The system is thoroughly documented through OpenAPI specification documentation to support implementation by third parties.

4. Results

Technical outcomes

The implementation achieved remarkable performance metrics with average response times ranging from milliseconds to seconds, depending on calculation complexity. The system operates successfully in parallel with the existing web platform without interference. Our serverless approach has proven cost-effective, with usage-based pricing that scales efficiently with demand. The system successfully supports all methodology versions, ensuring consistent calculations across different implementation requirements.

Integration Success

The API delivers structured input and output formats that align with industry requirements and expectations. A comprehensive error handling system provides clear, actionable feedback to API consumers. The system generates detailed calculation results across all emission scopes, enabling thorough carbon accounting. Version-controlled methodology access ensures calculations remain consistent and compliant with industry standards.

5. Conclusion

Key findings

Our implementation demonstrated that serverless architecture provides an optimal balance between cost and performance for carbon calculation services. We successfully adapted the existing calculation engine for API access without compromising accuracy or reliability. Version control proved essential for maintaining calculation consistency across different methodology implementations. The development of comprehensive documentation has emerged as a critical factor in facilitating third-party adoption.

Benefits to industry

The implementation enables seamless integration of automated carbon accounting into existing agricultural management systems. Organizations can now significantly reduce their reliance on manual data entry processes. The system supports efficient large-scale emissions reporting while ensuring consistent methodology application across different implementations. The API integration capability allows farm management systems to incorporate carbon calculations directly into their existing workflows, enhancing operational efficiency.

6. Future research and recommendations

Looking ahead, we identify several critical areas for future development. The ASP.NET Core API requires an upgrade from .NET 6 before March 2025 to ensure continued system reliability and security. We recommend aligning methodology versions with the latest PICCC GAF tools to maintain industry consistency. Future enhancements should consider implementing batch processing capabilities and exploring real-time calculation options. The development of additional integration examples would further support industry adoption.

7. References

The project drew upon several key resources including the PICCC.org.au GAF Tool Documentation, AWS Lambda and API Gateway Documentation, OpenAPI Specification Standards, and existing MLA Carbon Calculator Documentation. These resources provided crucial guidance throughout the development process.