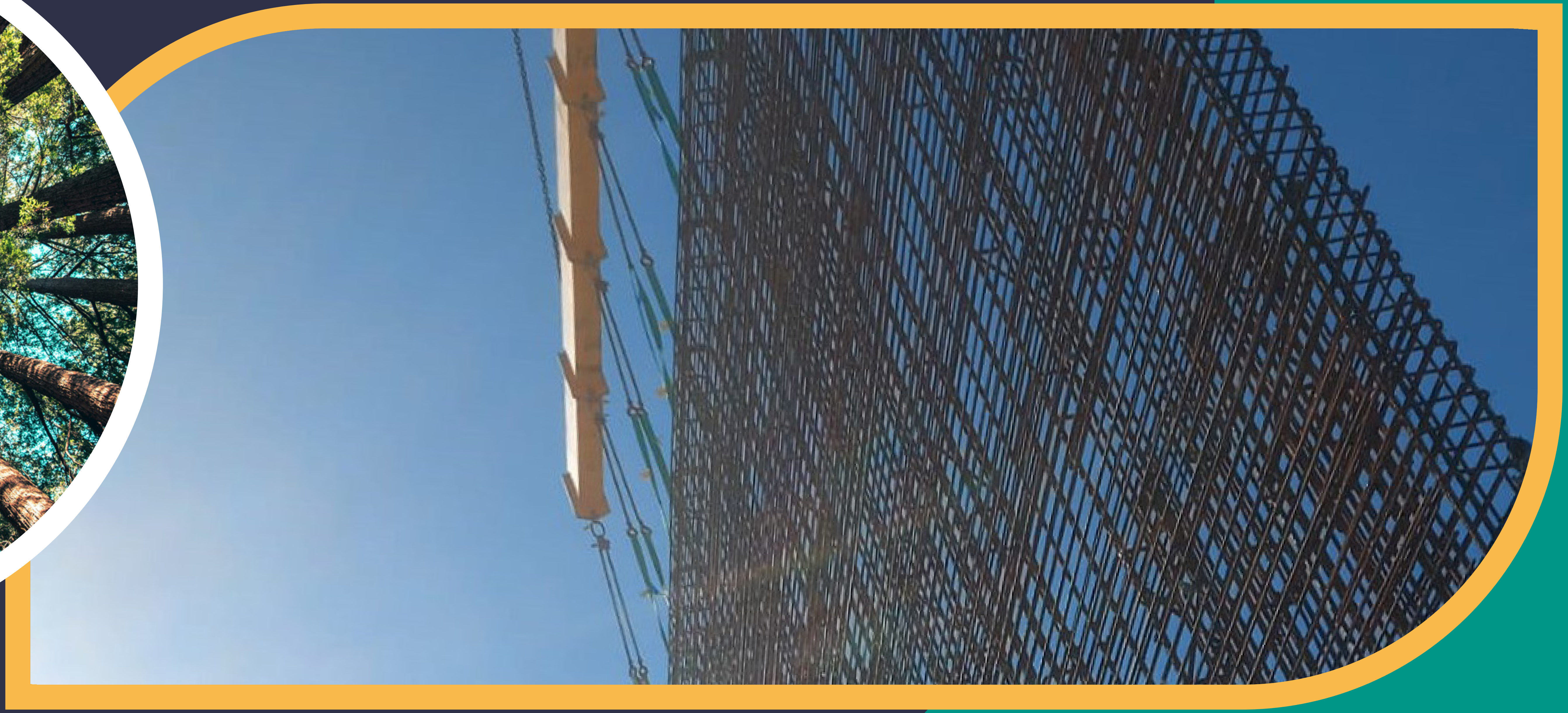
NightingaleHQ

Serverless Cloud Architectures for Sustainable Manufacturing Compute

Our approach to sustainability-first serverless cloud architecture

Authors: Chris Wilson, Ruth Kearney, Richard Jackson



Contents

Executive summary

This whitepaper outlines our approach and technological reasoning behind designing sustainability-first software solutions, with a focus on serverless cloud computing. We discuss key principles for carbon-aware software development, emphasising energy-efficient coding, serverless architectures, and code optimisation for energy efficiency. The proposed architecture, centred on Microsoft's Azure cloud platform, is detailed with insights into creating a scalable and secure solution for connecting to external systems. It highlights the use of Azure Logic Apps and Azure API Management, both employing pay-per-use pricing models, promoting cost efficiency proportional to usage. The Carbon Aware SDK is introduced as a tool to choose regions providing the most sustainable energy, aligning with principles of serverless design. We explore how to increase developer velocity as a metric for efficiency in software development, linking it to sustainability goals. We also present various development technologies, including Data API Builder (DAB) and Strawberry Shake, to increase developer velocity and contribute to sustainability targets. The paper concludes by emphasising the shift towards serverless computing as a key element in reducing environmental impact while increasing operational efficiency in manufacturing. Future work is outlined, including application within the steel manufacturing sector where the focus will be to minimise waste in rebar cutting and allocate computing resources in serverless cloud environments for maximum sustainability. In summary, the whitepaper provides an exploration of sustainable software development practices, technological tools, and a forward-looking perspective on the intersection of technology and carbon awareness in the manufacturing sector.

Background

The whitepaper stems from the project "Re-imagining Design & Planning for Greener Reinforcement Steel Manufacturing," funded by EIT Manufacturing and Innovate UK, and led by Midland Steel. Pan-European partners include steel manufacturer Bastal AS, Kuka Robotics, civil engineers O'Connor, Sutton, Cronin (OCSC) and research partners VTT. It focuses on reducing carbon emissions in steel manufacturing through standardisation, modular prefabrication, and advanced technologies like Building Information Modelling (BIM) and Artificial Intelligence (AI). The project aims to make all stages of rebar steel manufacturing greener, employing optimised planning and scheduling to reduce waste and associated carbon emissions. A new concept of standardisation facilitating scalable volumes is tested, which is produced offsite as standardised modular prefabricated sections reducing waste significantly. Reducing waste from cutting rebar is crucial for lowering greenhouse gas emissions and despite advancements in technology and processes, offcut (waste) still stands at 3 to 5% (Kwon & Kim 2021). Rebar is a key material in reinforced concrete, and it produces more carbon dioxide per unit weight than other construction materials. This overall objective aligns with global sustainability goals, particularly the Steel Breakthrough initiative for near-zero emission steel by 2030. On a global scale, for 2019 the estimated annual rebar cutting waste was 28.4 million tons with estimated carbon emissions of 9,7 million tons (Kwon & Kim 2021).This is no small environmental challenge and one where smarter technology choices and greener practices can play a significant role in carbon reductions or savings. As technology partners, our goal is to optimise the planning process of Cut & Bent production to deliver a reduction in waste and associated carbon emissions. Our planning optimisation solution will support 'the Planner' to generate results on more efficient ways to schedule jobs based on a set of parameters including inventory availability and energy requirements. The tool leverages industry standards and models, such as the ESMF Industry 4.0 Core Information Model and the IEC 62264-1. The model provides flexibility and extensibility to cover a wide range of manufacturing disciplines and serves as the basis for creating a generic model upon which our application is built. It is the foundation for which we will extend functionality for our steel production usecase.

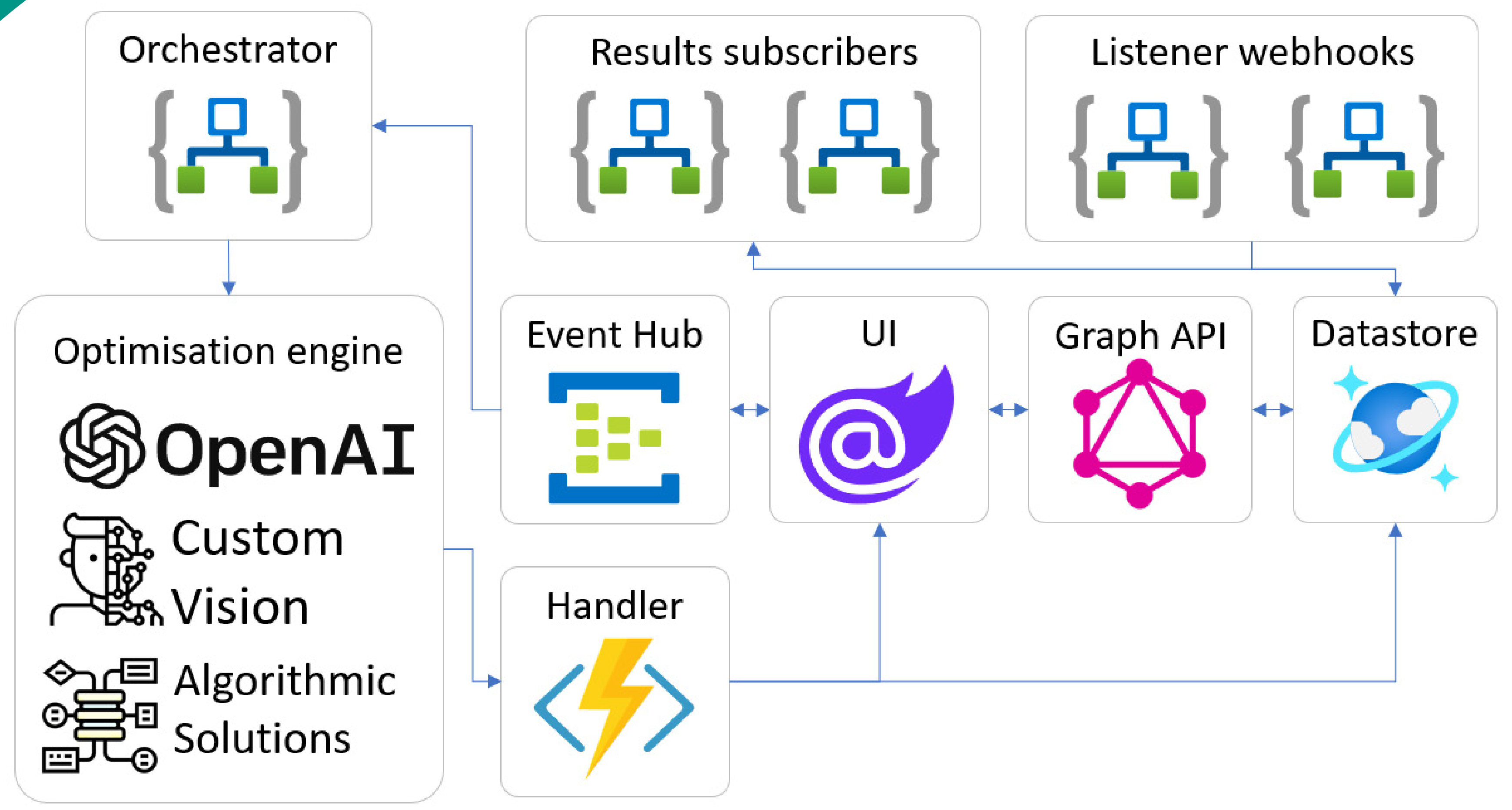


Figure 2 Generic serverless architecture as utilised by Nightingale HQ

1. Developing a serverless computing architect

Here we present our serverless computing architect designed from a sustainability-first perspective. With multiple cloud service providers offering a diverse range of compute resources, it is essential that cloud solution architects are aware of sustainability practices and implement them accordingly. Our architecture adheres to key principles of serverless design and can be used as the basis for building sustainable manufacturing solutions. It is designed to be scalable, secure, and cost-effective. We use a combination of Azure services, including Azure Functions, Azure Cosmos DB, Azure API Management, Azure Logic Apps, and a containerised section for utilising data processing compute, such as OpenAI. Containerisation allows for scaling of compute resources while also enabling robust security ("sandboxing") of data, thus providing governance and control of what data is accessible by the compute. Scaling provisions with more resources when the demand is high, and fewer resources when the demand is low, which allows us to minimise waste by only using the resources that are needed. The compute required for data analysis and training of machine learning models can be vast and may require expensive High-Powered Compute (HPC) solutions. By using a containerised approach, we can scale up or down as needed while isolating HPC operations, which can help reduce costs and make the solution more cost-effective. Additionally, this also means we do not need to operate our own HPC clusters but can rent time from shared infrastructure operated by Microsoft. To understand our approach to the technical architecture, first we cover some of the core concepts followed by an exploration of the role of greener software within sustainable manufacturing.

Green software and sustainable manufacturing

Green software is a strategic approach aimed at minimising the carbon footprint of digital systems. It achieves this by optimising code for energy efficiency, utilising renewable energy sources for server power, and minimising data transmission over networks. This approach, known as transparent efficiency, does not necessitate changes in user behaviour and thus enables a swifter rollout with quicker realised benefits and often lower costs. Organisations like the Green Software Foundation and European Green Digital Coalition (EGDC) are pioneering methodologies to estimate the environmental impact of digital solutions across various sectors. Initiatives like these are critical in measuring both the direct footprint of a digital solution and its positive contributions, such as carbon reduction.

Our solution of process optimisation aims to minimise negative environmental impacts as outlined in the EU Sustainability Guide. Given that rebar, a fundamental resource in reinforced concrete structures, has a higher carbon dioxide output per unit weight than other construction materials, reducing rebar cutting waste directly contributes to greenhouse gas (GHG) reduction. We aim to achieve this reduction by implementing an optimisation process that saves through efficiency improvements, employing a green architecture to minimise long-term impact, and utilising green development practices to reduce the overall impact of the development process. We are entering into an era where the adoption of greener software principles is a strategic move by manufacturers to enhance sustainability. 65% of manufacturing executives now recognise sustainability as a top-tier concern according to global consultancy CIMdata and the alignment between sustainability and profitability is also evident. Manufacturers that are incorporating sustainable practices are reported to outperform their counterparts by 21% in both profitability and positive sustainability outcomes as reported by the UN Global Compact survey. These findings are supported by IBM's 2022 CEO Study indicating that businesses implementing sustainability and digital transformation initiatives, including green coding and software development, report a higher average operating margin than their peers. This all underscores the strategic importance of embracing greener software principles in the changing industrial environment.

What role can greener software play?

As manufacturers strive to become more sustainable, they are not only examining their established business processes for greater efficiency but also demanding upstream and downstream efficiency improvements from their partners and suppliers. Developing software under greener software principles is a strategy aimed at minimising the overall environmental impact of technology. This approach involves reducing the carbon footprint of all technical operations, from manufacturing line data processing and datacentre usage to the human cost of developing solutions. Carbon aware computing encompasses not only green coding but also green software, which refers to applications developed using environmentally friendly coding practices. We will now examine these key concepts in closer detail and detail their application within our manufacturing usecase.

Carbon-aware software

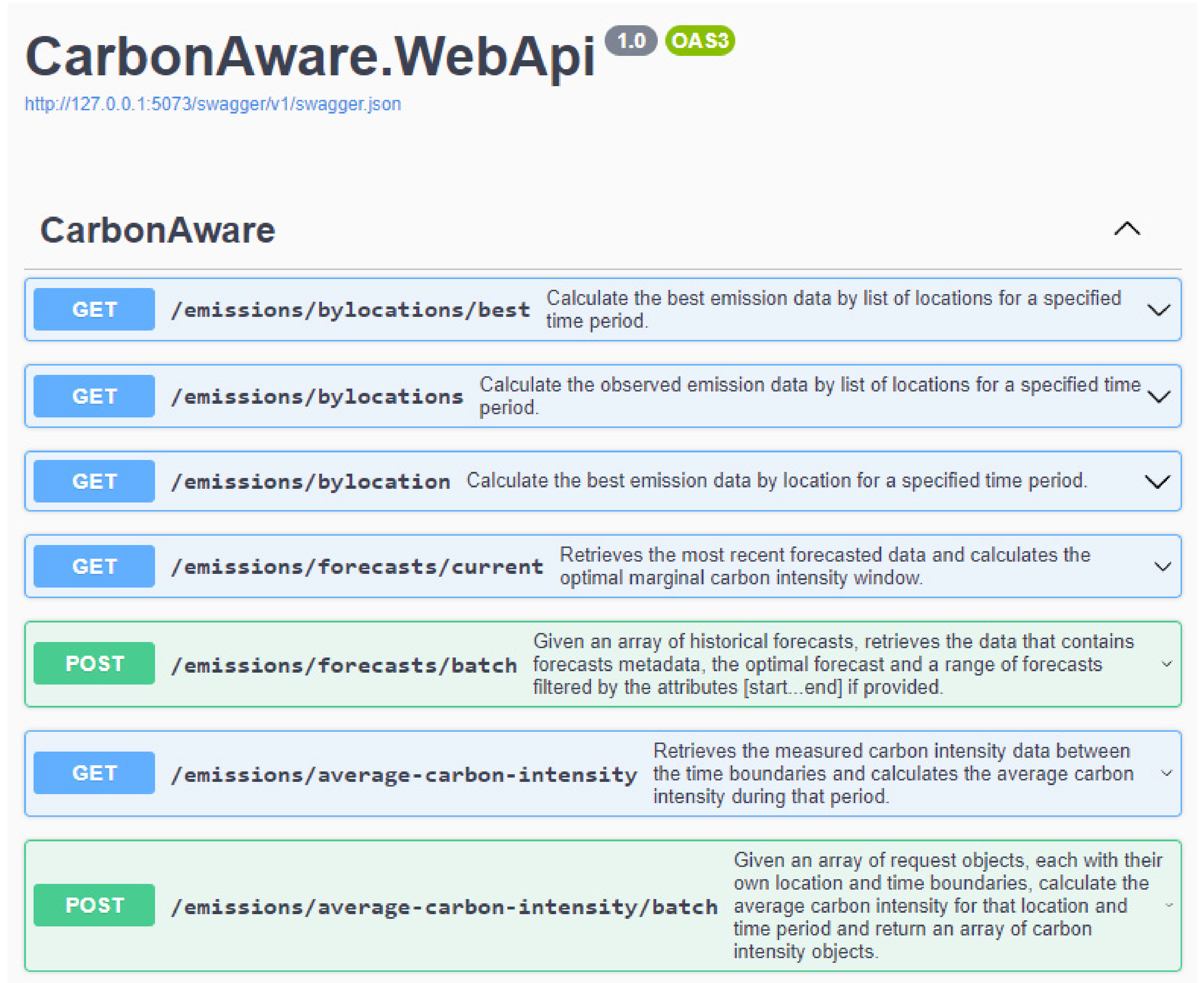
The Green Software Foundation defines the principle of Carbon Awareness as "do more when the electricity is cleaner and do less when the electricity is dirtier". Microsoft's white paper on Carbon-aware computing identifies three key principles for governing application behaviour to compliment Carbon Aware software design:

- **Time-shifting:** optimise impact by shifting workloads based on the Location-Based Marginal Carbon Intensity prediction of the data center
- **Location-shifting:** optimise impact by shifting workloads based on the Location-Based Marginal Carbon Intensity prediction of a location
- **Demand-shaping:** running software so it does more when electricity is clean and less when it is dirty

The outcome of these three factors is the running of software in the best place, at the best time, with the best energy possible, to reduce the carbon footprint of the software's processes.

The carbon footprint of software?

When using cloud computing services, it is important to consider the carbon footprint of the software itself, as well as the carbon footprint of the cloud provider. The carbon footprint of the software can be measured by considering factors such as the energy consumption of servers, the energy used to transmit data over networks, and the energy used by end-user devices. Tools such as Cloud Carbon Footprint can help developers estimate the carbon footprint of their applications. This tool considers factors such as the energy consumption of servers, the energy used to transmit data over networks, and the energy used by end-user devices. Major cloud service providers also provide tools that can help developers estimate the carbon footprint of their applications, such as AWS Customer Carbon Footprint Tool and Azure Emissions Impact Dashboard. The Carbon-Aware SDK is an open-source software tool for identifying the current emissions outputs for data centres and regions. The tool provides a set of REST API endpoints for developers to query, which can then be used to make decisions around the optimal location software should be run. With recommendations and data presented to the user, a manual choice could then be made to determine which location a user wishes to run a task, or the decision-making could be fully automated, so that regardless of user-preference the software will always run in the recommended location.


CarbonAware.WebApi 1.0 OAS3
http://127.0.0.1:5073/swagger/v1/swagger.json

CarbonAware

GET /emissions/bylocations/best Calculate the best emission data by list of locations for a specified time period.

GET /emissions/bylocations Calculate the observed emission data by list of locations for a specified time period.

GET /emissions/bylocation Calculate the best emission data by location for a specified time period.

GET /emissions/forecasts/current Retrieves the most recent forecasted data and calculates the optimal marginal carbon intensity window.

POST /emissions/forecasts/batch Given an array of historical forecasts, retrieves the data that contains forecasts metadata, the optimal forecast and a range of forecasts filtered by the attributes [start...end] if provided.

GET /emissions/average-carbon-intensity Retrieves the measured carbon intensity data between the time boundaries and calculates the average carbon intensity during that period.

POST /emissions/average-carbon-intensity/batch Given an array of request objects, each with their own location and time boundaries, calculate the average carbon intensity for that location and time period and return an array of carbon intensity objects.

Figure 1 Carbon Aware SDK, showing API endpoints for requesting data on areas such as emissions by location and average carbon intensity

The Green Software Foundation also maintain the Software Carbon Intensity (SCI) specification which details the calculations needed for software application carbon intensity scores. This allows companies and developers to implement their own tooling, utilising these calculations and provides complete transparency as to how these figures were generated. Understanding these results, and what factors have the largest impact on each use case, is essential for understanding where change needs to be implemented to improve these results. The documentation for the Carbon Aware SDK gives the following use case example: "Machine Learning (ML) workloads are a great example of long running compute intensive workloads, that often are also not time critical. By moving these workloads to a different time, the carbon emissions from the ML training can be reduced by up to 15%, and by moving the location of the training this can be reduced even further, at times by up to 50% or more."

What does it mean to be carbon-aware?

Carbon-aware software is an emerging discipline at the intersection of climate science, software, hardware, energy markets, and datacentre design. Carbon-aware software refers to applications that adjust their operations based on the carbon intensity of the electricity supply. These applications work harder when the electricity is clean (generated from greener sources like wind and solar) and work less when the electricity is dirty (produced by burning fossil fuels) to reduce the carbon footprint. The concept of carbon-aware computing is a foundational principle of Green Software. The vision of green software is to build software that has no harmful effect on the environment. This can be achieved through reducing energy usage, using less hardware, and modifying computation to take advantage of the lowest-carbon sources of energy possible.

2. Serverless computing and the cloud

The synergies between sustainable manufacturing and green software are essential for creating an integrated approach to environmental sustainability and addressing the challenges posed by both physical and digital aspects of production. In this next section, we look at the concept of designing for serverless computing and how it can reduce the carbon footprint in manufacturing by only using resources when required. We also explore High-performance Computing (HPC) as a way for manufacturers to run core tasks such as scheduling, queueing, and executing workloads in a more sustainable way. Cloud-native technologies enable businesses to work towards the goals of sustainable manufacturing by enhancing efficiency, reducing environmental impact, and supporting overall sustainability goals within the manufacturing sector. These technologies offer scalability, flexibility, and cost-effectiveness. In the context of sustainable manufacturing, cloud-native solutions can play a pivotal role in optimising processes and resource utilisation, which is the core thesis of our work; to optimise production to reduce waste and to do so in a more sustainable way.

Cloud-native solutions

The term "Cloud Native" means software designed to be implemented in a cloud infrastructure, using the resources and compute provided by cloud software providers such as Microsoft Azure, Amazon Web Services and Google Cloud. The range of resources and compute available through such cloud software providers are vast, providing developers a wide range of different approaches and architectures, developed around the building blocks provided. One such approach is known as "Serverless computing".

Serverless computing

In traditional cloud computing models, the business is responsible for managing servers and scaling applications. This can be a time-consuming and expensive process, as it requires the business to hire and train staff to manage servers, as well as purchase and maintain hardware.

> ***Serverless computing* is a cloud computing model in which the cloud provider manages the allocation and provisioning of servers. This allows the business's developers to focus on writing code without having to worry about managing servers or scaling applications.**

For cloud computing "waste" can include the provisioning of resource which are using active but not being used - a computer which is on but is not being used is more wasteful than a computer which is only turned on when it is needed. Serverless computing can help reduce waste by only using resources when they are needed, which can help reduce costs and make serverless architectures more cost-effective. A key benefit of serverless computing is that it can reduce costs, as the business only pays for the resources that are used. This can be particularly beneficial for businesses that have fluctuating workloads, as they can scale up or down as needed without having to pay for unused resources.

Implementing a cloud framework

When designing a cloud infrastructure, it is important to be mindful of differences to on-premises (known as on-prem), infrastructure. Due to the nature of on-prem servers being dedicated to your business's operations, this means that there are considerable running costs as this compute is running with
high availability constantly. Using cloud infrastructure allows for the use of serverless compute, which will only be running at the point it is required.

Cloud software providers often provide guidance on the best practices for architecting on their platforms such as Amazon's "AWS Well-Architected", Google's "Google Cloud Architecture Framework" and Microsoft's "Well-Architected Framework". This guidance will typically be to define a set of guiding tenets that you can use to improve the quality of a workload, around topics such as reliability, security, cost optimisation, operational excellence, performance efficiency and so on. It is important for any software architect to familiarise themselves with the guidance of their chosen platform and to be aware of any changes and updates made to them as more resource offerings are delivered.

Key principles of serverless design

Here are five key principles to follow when designing serverless solutions in Azure:

1. **Event-driven:** serverless architectures are event-driven, meaning that they only execute code in response to specific events or triggers, such as a user request or a change in data.

2. **Optimise resources**: to reduce costs and improve sustainability, it is important to optimise the resources used by serverless functions. This can include using smaller function sizes, reducing the amount of memory allocated to functions, and minimising the number of function invocations.

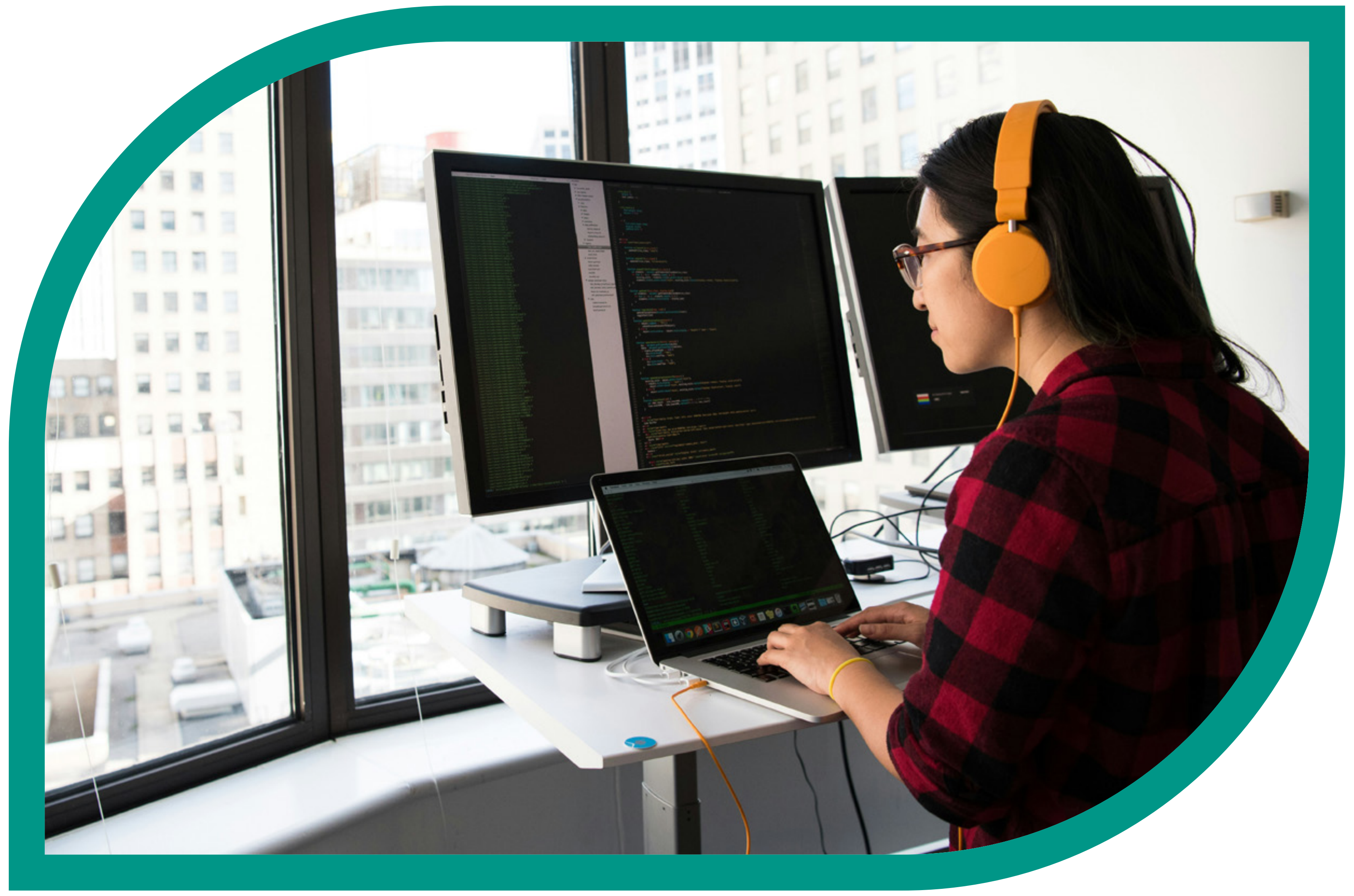
3. **Stateless:** serverless architectures are stateless, meaning that they do not maintain any persistent state between function invocations. Instead, they rely on external storage services, such as databases or object stores, to manage state.

4. **Scalable:** serverless architectures are highly scalable, as they can automatically scale horizontally and vertically based on demand. This can help reduce waste and energy consumption.

5. **Pay-per-use pricing:** serverless computing platforms use a pay-per-use pricing model. This means that developers only pay for the resources that their functions consume, which can help reduce costs and make serverless architectures more cost-effective.

Serverless computing and sustainable manufacturing

When developing software for sustainable manufacturing, it is important to consider the environmental impact of the software itself, as well as the environmental impact of the manufacturing process. Serverless computing compliments the goals of sustainable manufacturing through allowing manufacturers to reduce their carbon footprint by only using resources when they are needed. This can help reduce energy consumption and lower the overall carbon emissions of the manufacturing process. By leveraging serverless computing, manufacturers can create more sustainable and cost-effective manufacturing processes.



Considerations

Here are five considerations when implementing carbon-aware solutions:

1. **Energy-efficient coding strategies:** using energy-efficient coding strategies, such as minimising network requests, optimising algorithms, and using caching, will reduce the energy consumption of their applications.

2. **Clean energy sources:** making conscious choices about where and when your applications are run through tools such as Carbon Aware SDK.

3. **Data transmission:** reducing the amount of data that needs to be transmitted over networks. This can include using compression algorithms, minimising network requests, and using caching to reduce the amount of data that needs to be transmitted.

4. **Carbon reporting:** developer mechanisms to report on the carbon emissions associated with running an application. This can help developers identify areas where they can reduce the carbon footprint of their applications.

5. **Scalability:** carbon-aware solutions should be designed to be scalable, so that they can handle increases in demand without consuming excessive amounts of energy.

3. High performance computing (HPC)

High-performance computing (HPC) is the term for the compute used during processes that require vast amounts of compute to operate, such as digital twin usage, large-scale data operations or AI model training. HPC uses clusters of powerful processors, working in parallel, to process massive multi-dimensional datasets (big data) and solve complex problems at extremely high speeds typically more than one million times faster than the fastest commodity desktop, laptop, or server systems. Due to the size and costs of maintaining high-performance compute, cloud service providers offer the usage of these compute when required, allowing for HPC usage without the internal investment of purchase and maintenance. HPC focuses on aggregating compute in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order. Cloud service providers will typically provide a resource for running large-scale HPC applications, such as Amazon's AWS Batch, Google's Batch and Microsoft's Azure Batch. These resources will give you functionality such as scheduling, queueing, and executing workloads on HPC compute, giving the developers control over HPC usage.

HPC in manufacturing

High-performance computing (HPC) can significantly benefit the manufacturing industry in three key ways:

1. **Optimisation:** manufacturing processes are complex and involve numerous stakeholders each with various concerns and dynamic priorities. HPC-driven AI can help engineers and operators optimise these processes.

2. **Advanced simulations:** HPC can help run advanced design simulations, utilise, and support digital twins, automate processes, and predict maintenance issues. This can boost manufacturing performance and reduce time to results.

3. **Data analytics:** with products increasingly connected and generating vast amounts of data, HPC can provide the large computing resources needed on-demand to process and analyse this data and uncover patterns.

HPC clusters can be extremely expensive to setup and are impractical for all but the largest enterprises to consider deploying, as a result smaller organisations may still benefit from HPC but will need to rent time in shared infrastructure deployed by a company such as Microsoft, or by using models that have already been trained in a HPC cluster

AI and AI models

The computational complexity of AI processing varies depending on the model being used, for example; the GPT-3 language model has 175 billion parameters and requires 355 years of compute to train on a single GPU, GPT-2 by comparison has 1.5 billion parameters and requires 7.5 months of compute to train. While making for a simpler comparison, no model would be trained on a single GPU but rather using a cluster of custom GPUs. The GPT-3 model was trained on 10,000 GPUs which would reduce the training time to several days. However, this is still a considerable amount of compute, and the carbon footprint of training such a model is considerable - some industry experts estimate this could be up to 200,000 kg CO2.

The carbon footprint of inference (using the model to validate a parameter) however is much lower as the model has already been trained and is simply being used to process data. This is important when considering whether the carbon cost of training a custom model is worthwhile as pre-trained models have already spent their carbon budget during the training phase and can now be used more widely, potentially lowering their overall carbon footprint. Custom models may be too limited to use more widely and may incur a higher implicit carbon cost. Due to the considerable compute required for AI tooling to run, HPC is often used for processing AI and machine learning algorithms. Intel notes that "The main driver for HPC and AI workloads is the persistent growth of data and the need to match pace with HPC-scale analysis." - as the quantity of data available continues to rise, and the complexity of our potential querying grows alongside that, so does the need for the compute to handle these queries. HPC satisfies this need. However, owing to HPC's potential for vast energy usage, such operations can have considerable carbon footprints. To that end it is important to fully understand the specific requirements of the task or model being processed by the HPC, and to choose the appropriate AI models for the task.

Containerisation using HPC and AI

The resource running the HPC can be containerised. This holds numerous benefits, primarily that of scalability and resource management. As HPC and AI applications can take significant resource, containerisation allows for the applications to be scaled up or down based on demand. Containers can be easily deployed and orchestrated across a cluster of machines, allowing for efficient utilisation of resources and improved performance.

Using HPC in a sustainable way

High-performance computing (HPC) can be utilised sustainably through various measures. Prioritising energy efficiency is crucial, and the use of public clouds that emphasise renewable energy can help mitigate the substantial electricity demands of HPC. In addition, as we have highlighted previously, the carbon footprint of datacentres, influenced by the energy production's environmental impact, can be minimised by selecting locations with a lower carbon footprint. It is also important that sustainable design in Hardware Lifecycle Management is essential, encompassing considerations from chip manufacturing to semiconductor recycling. Finally, HPC can contribute to the development of a circular economy if we set the criteria in a way that supports sustainability to its full potential. This could include 'recycling' the supercomputer locations. Overall, HPC offers promising outcomes for a more sustainable future, however it involves careful consideration of energy sources, location selection, hardware lifecycle, and adherence to principles of the circular economy.

4. Our approach to sustainability-first serverless cloud architecture

We have designed a serverless cloud architecture serving as a foundation for developing sustainable manufacturing solutions. This section looks at the technology decisions taken to build the architecture and considerations into a green approach to software development.

Generic architecture

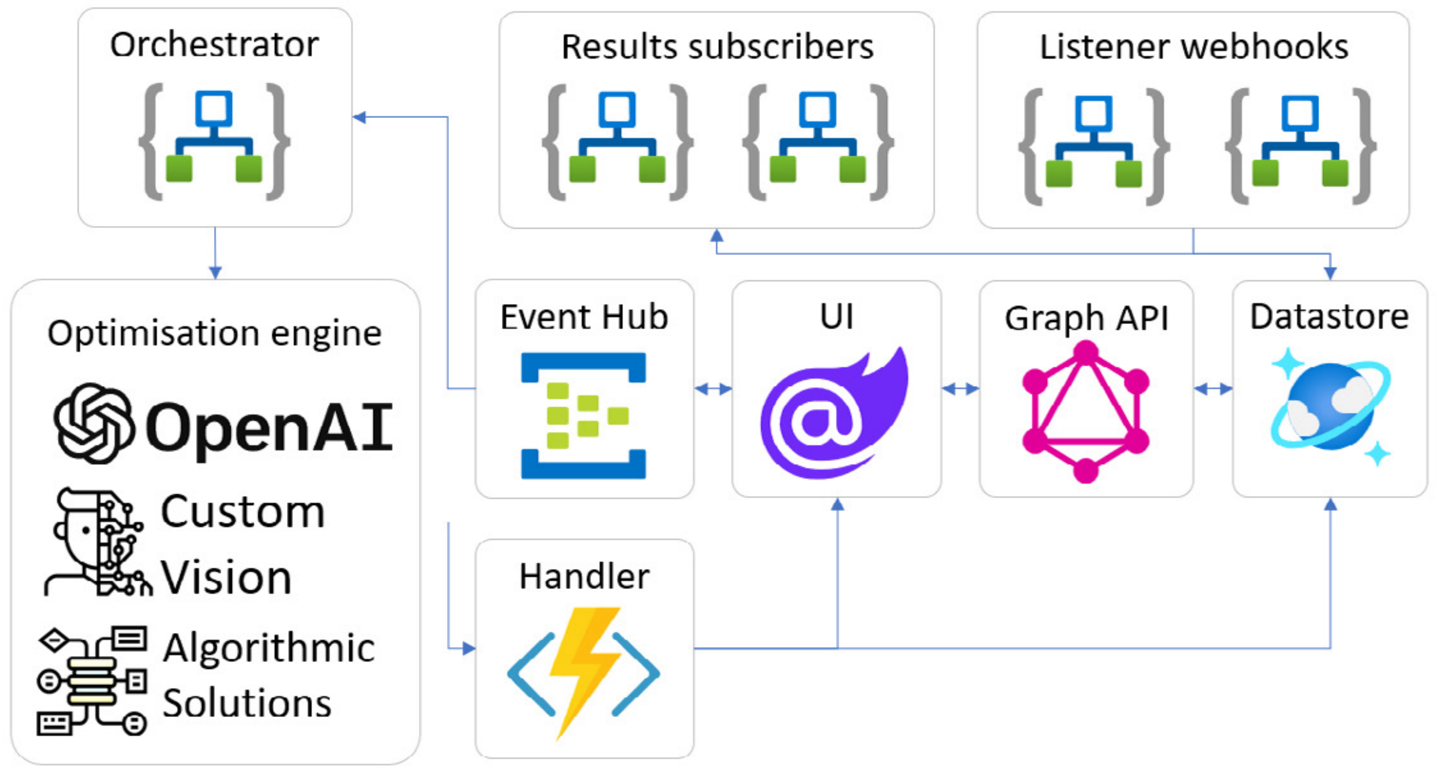


Figure 2 Generic serverless architecture as utilised by Nightingale HQ

Overview of architecture

The architecture is designed to be scalable, secure, and cost-effective. It uses a combination of Azure services, including Azure Functions, Azure Cosmos DB, Azure API Management, Azure Logic Apps, and a containerised section for utilising data processing compute, such as OpenAI.

The UI app for user interaction is hosted within a Static Web App and will typically be a Single Page Application. Requests are made via the Graph API, which can query the Azure Cosmos DB or send a request to the Azure Event Hub. The Azure Event Hub then sends the request to the containerised compute. The results from this compute are then sent back to the Graph API, which can then be sent back to the user. This architecture also allows for the creation of a Subscriber and Publisher API endpoints, which can be used to connect to external systems through a combination of Azure Logic Apps and Azure API Management.

Event-driven architecture

Event-driven architectures are a key component of serverless computing, as they allow for the execution of code in response to specific events or triggers. This allows us to only spool up resources when they are needed and thus reduce not only running costs but the associated carbon footprint of the application. For example, as we use Azure Cosmos DB as our database provider, we can use Cosmos DB's change feed functionality to power our event-driven architecture, which is then used to drive processing workflows with Azure Functions. Similarly, we can spool up compute resources to handle data analysis when a request is received, and then shut down the compute resources when the request has been fulfilled.

A secondary advantage of this approach is that it allows for the decoupling of the application's components, which can help improve scalability and reliability. For example, if the application's processing hub goes down, the application can still cache processing requests if the event hub is still running - meaning the user experience is not impacted.

Client-side application frameworks

Client-side application frameworks such as Blazor WASM (Web Assembly) allow for the creation of Single Page Applications (SPAs) which can be hosted within a Static Web App. This allows for the creation of a user interface which can be hosted within a Static Web App, which can then be used to interact with the Graph API - all without requiring a costly server to run constantly and while still allowing us to design an app which works from the API-first principle. The secondary advantage is that this allows the user to run the application on a low-powered device, such as a mobile phone, which can help reduce the carbon footprint of the application as the device is not using as much energy as a desktop or laptop computer, or more importantly, a web server.

Provisioning resources

Many of these resources can be provisioned upon request, utilising the core principles of serverless compute. This allows for a reduction in waste and costs as the resources do not need to be provisioned until they are required. While all resources could utilise a dynamic "upon request" resource provisioning strategy, there are specific resources which would impact on the user experience if they were not always available - specifically the database and the front-end app. These resources would be required to be pre-provisioned to ensure that the user experience is not impacted. Longer processes, such as data analysis, which would typically take a long time to run, can be provisioned upon request, as the user is not expecting an immediate response. This balance allows for a positive user experience whilst minimising waste.

Dynamic provisioning

Dynamic provisioning is a demand-based allocation strategy that allows for the creation of resources only when they are needed. This can help reduce waste and costs, as resources are only created when they are needed. This approach lends itself well to both scalability and sustainability, as it allows for resources to be created on-demand, which can help reduce waste and costs. This technique is traditionally applied both horizontally (adding more instances) as well as vertically (increasing the capability of instances) but for our purposes we will be thinking in another vector - geographically. By using serverless resources and executing them in the most sustainable location, we can reduce the carbon footprint of our software.

Usage of Containerisation

Containerisation is used for all compute focused on data analysis, such as through algorithmic solutions, machine learning or OpenAI. Containerisation allows for scaling of compute resources while also enabling robust security ("sandboxing") of data, thus providing governance and control of what data is accessible by the compute. Scaling provisions more resources when the demand is high, and fewer resources when the demand is low, which allows us to minimise waste by only using the resources that are needed. The compute required for data analysis can be vast and may require High Powered Compute solutions, which can be expensive to run. By using a containerised approach, we can scale up or down as needed, which can help reduce costs and make the solution more cost-effective.

Usage of serverless compute

The core user interface is hosted within a Static Web App as it allows for the deployment of a website or web application without the need for a server to be running constantly. This means that the website or application can be served directly from a Content Delivery Network (CDN).

CDNs typically implement an edge-network approach to content delivery where servers are placed as close to users as possible, this has the effect of decreasing loading times for users but also has the combined effect of using less network equipment to transmit the data and reduces the communications carbon footprint.

Infrastructure used to host the CDNs is also shared between multiple providers and users, which further reduces the carbon footprint of the solution as compared to keeping a server running hot just in case a user request is received. Azure's Static Web Apps can be deployed either through a pay-per-use pricing model or as static, reserved instances. The latter are more expensive but provide a guaranteed level of service at the cost of a higher carbon consumption as the architecture is not shared and thus may be underutilised when the application is not experiencing throughput. As a result, we will not be taking this approach and will instead be using the serverless, shared pay-per-use model while monitoring usage closely. Should application throughput levels tend towards unsustainable levels we will re-assess this balance. Depending on how you implement them it is possible to run serverless functions for next-to-nothing, or in some cases even on the Azure free tier, as is evidenced by the widely integrated password safety platform Have I Been Pwned (HIBP) Case Study: Doing Big Things for Small Dollars. The key challenge is to identify where your costs originate and to try and minimise or negate that impact. In HIBP's case they expected high utilisation of common hash queries on their API and Azure charges for outgoing data, so they implemented Cloudflare workers to cache common requests and even handle OPTIONS responses. While this is a long-term goal for us as we grow, we will be focussing on offloading processing from the shared infrastructure in the near term - Static Web Apps and Graph caching will help us in this regard.

To provide connectivity to external systems via results and listener APIs, a combination of Azure Logic Apps and Azure API Management is used. Both tools work together to provide a scalable and secure solution for connecting to external systems.

Azure Logic Apps provides a serverless workflow engine that can be used to orchestrate the flow of data between systems, while Azure API Management provides a gateway for exposing APIs to external systems. By using these tools together, we can create a scalable and secure solution for connecting to external systems. It should also be highlighted that Azure Logic Apps and Azure API Management both use pay-per-use pricing models, which means that costs are proportional to usage and waste is minimised. This architecture can be created in multiple regions, and upon request the region which provides the most sustainable energy at that point can be chosen through the usage of the Carbon Aware SDK. This approach utilises the principle of Carbon-awareness within our software usage, allowing us to make sustainable decisions about the locations and energy our application is using.

Adhering to key principles of serverless design

Our architecture design follows the key principles defined below:

1. **Event-driven:** Azure Cosmos DB's change feed functionality can be used to power event-driven architectures, which is then used to drive processing workflows with Azure Functions (Source).

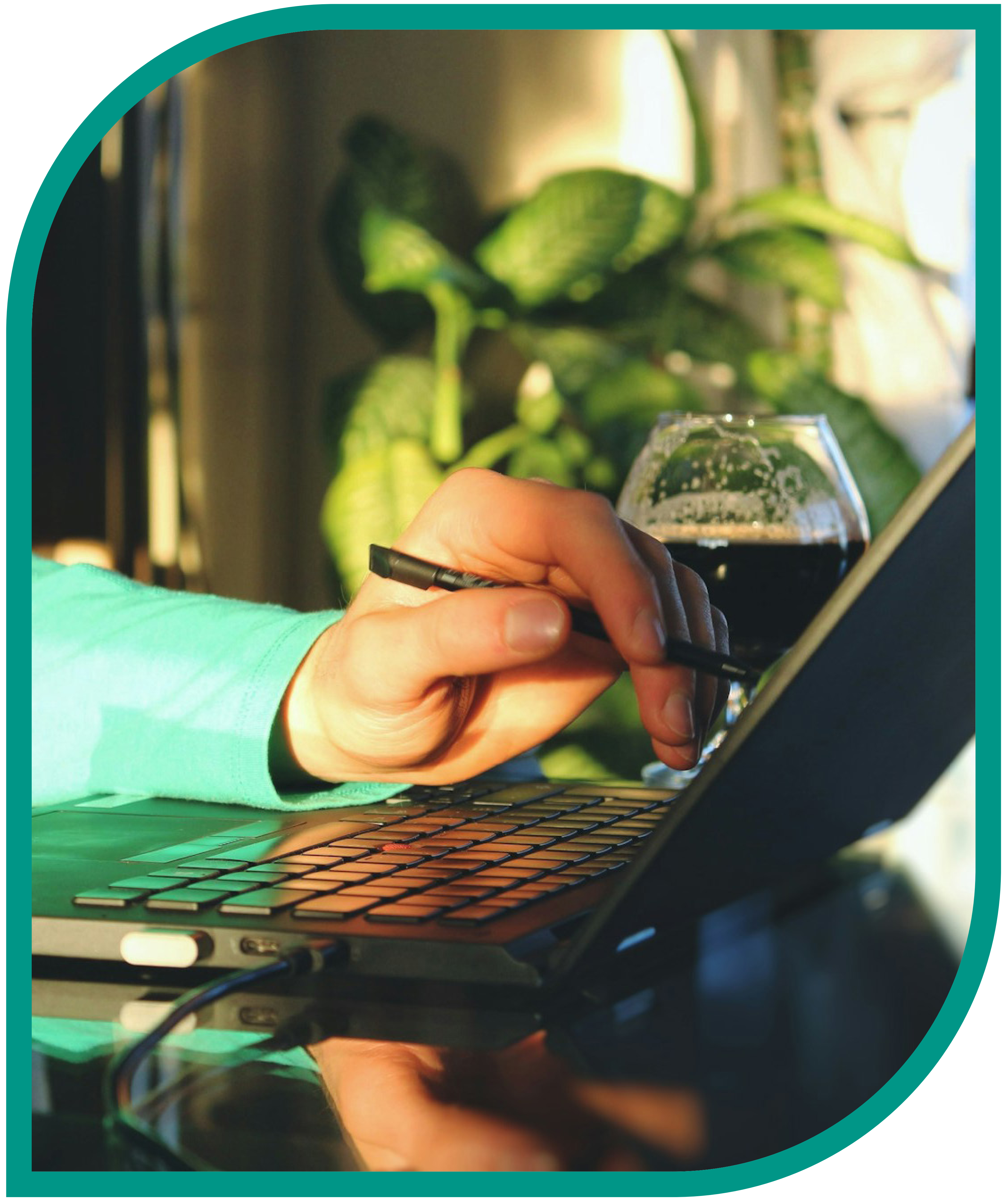
2. **Stateless:** all state is maintained by Azure Cosmos DB, with no other parts of the system storing state or being aware of or referencing previous transactions.

3. **Scalable:** all Azure resources used can scale in response to usage volume.

The architecture itself leans heavily on resources with pay-per-use pricing, ensuring that the costs are always proportional to the usage. The Cosmos DB containers must be organised in such a way as to reduce cross-container reads, as these can lead to quickly spiralling read charges.

Remote work

Remote work can result in an overall reduction in energy emissions by reducing the need for commuting and business travel. This can be achieved by using video conferencing tools such as Microsoft Teams, which can be used to conduct meetings remotely. This also means that a business operating entirely remotely can reduce the need for office space and in particular office environmental controls, which can further reduce energy emissions. Modern work is a new concept that is emerging because of the digital transformation of the workplace. It is a term that describes the way in which people work today, and how technology has changed the way we work. Modern work is characterised by a shift from traditional office-based work to a more flexible and mobile work environment. This shift has been enabled by the development of modern technologies such as cloud computing, mobile devices, and social media. These technologies have enabled people to work from anywhere at any time, which has led to an increase in remote working and a decrease in the need for office space.



5. Driving developer velocity

Developer velocity is a widely understood metric for how quickly developers can deliver new features and functionality. McKinsey & Company define developer velocity as:

> "Improving business performance through software development comes down to empowering developers, creating the right environment for them to innovate, and removing points of friction. Industry leaders refer to this capability as Developer Velocity."

It is the ability to drive transformative business performance through software development and is a key metric for measuring the efficiency of a development team and can be used to identify bottlenecks in the development process. Developer velocity is also a key metric for measuring the sustainability of a development team, as it can be used to identify areas where developers are spending too much time on non-value-added activities. For example, if developers are spending too much time on manual testing, this can be a sign that the team is not using automated testing tools effectively, or if developers are spending large periods on frontend component development, this may be a sign that application of a component library would be more efficient. We believe that increasing developer velocity can play a key role in developing greener solutions and it is in-part, what we are trying to do in practice by influencing the efficiency, speed, and agility of the software development process.

Developer Velocity for sustainability

Increased developer velocity has a direct impact on sustainability targets during development as it can directly reduce the amount of time spent on development, which in turn reduces the amount of energy consumed during development cycle. There are numerous industry approaches which aim to increase developer velocity, but they all widely achieve their aims by removing roadblocks and automating processes which would otherwise take time if handled manually. One such example is application deployment; instead of users having to manually deploy their projects, they can use a tool such as Azure DevOps to automate the deployment, validation, and testing process - this can be handled by a low-power build and deploy server (often hosted in shared architecture and thus enjoying the benefits of reduced energy consumption we detailed earlier in the document) rather than the developer's higher powered development machine, and in the vast majority of cases executes faster than if the developer handled this manually. This is an implicit carbon saving and contributes towards the project's sustainability goals. We have targeted several development technologies which we have identified as having the potential to increase developer velocity, they will be utilised by all developers on the project, as well as those who will be onboarded as the project progresses and are detailed in the following sections.

Data API Builder (DAB)

Data API Builder (DAB) is an open-source tool created by Microsoft which aims to help increase developer velocity by automatically generating commonly used REST and GraphQL database APIs. The endpoints created cover standard "CRUD (Create, Read, Update, Delete)" (create, read, update, delete) operations which are typical building blocks for any database usage within software development. It achieves this by exposing the objects defined in your database schema; this should realistically mean the time to adapt to new data structures or changes to existing structures is dramatically reduced, as well as removing some of the more monotonous code developers traditionally must write.

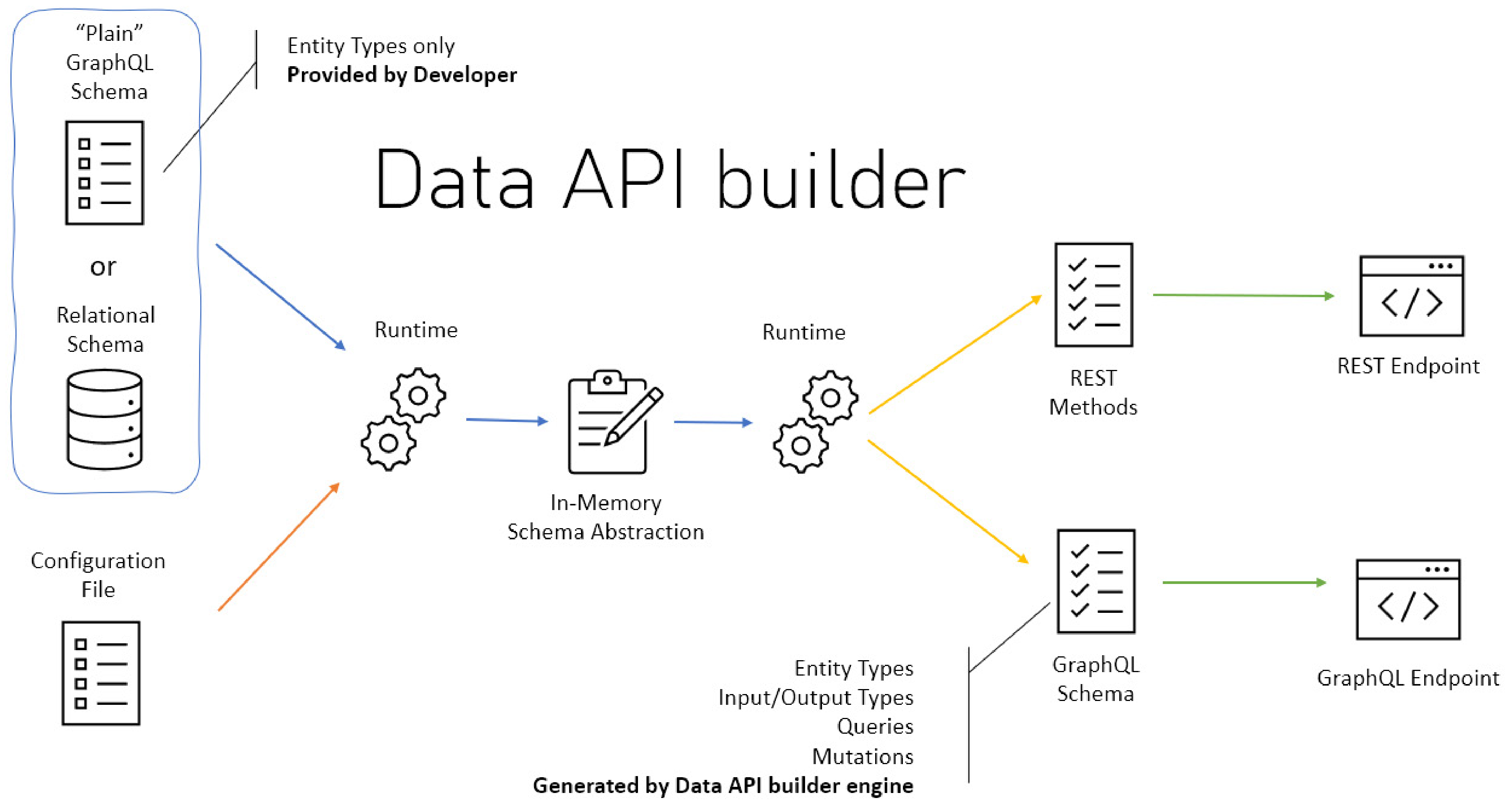


Figure 3 Technical diagram of Microsoft's "Data API Builder". Source: "Overview to Data API Builder"

We believe this approach will provide benefits to the data storage and modelling strategy we have employed as we anticipate frequent refactoring in the initial stages while we work out the best NoSQL structure for both the interfaces and optimisation bundles. Traditionally this would mean manually adding CRUD and query endpoints to the API every time a new database container is added, or updating the models should the properties be changed - even with tools like QuickType this is still largely a manual and tedious process which is prone to human error. DAB speeds this process by only requiring the developer to control the Entity Type definitions from which the subsequent .NET types are built. Based on a rough estimate from prior projects it takes a developer a week to create a simple API representing the models and queries required by a small application, this includes researching the database schemas, writing, and testing the endpoint logic, and writing a .NET client to consume it. Amending a single endpoint to represent a significant logic or model change takes half a day to a day. Through using DAB, we have found a developer can create an API in a matter of hours and amend endpoints in minutes. If realised, these savings not only represent a reduction in time-to-result but also an implicit carbon saving during the development process.

Strawberry Shake

We also aim to extend benefits gained from increased velocity by utilising GQL Client generation capabilities provided by Strawberry Shake. By doing so we eliminate the modelling and mapping process usually associated with API development and instead allow the developer to focus on the business logic of the application. This can vastly reduce the amount of time spent on the menial aspects of application development, again providing an implicit cost saving. When considered in the context of a NoSQL platform - where the natural evolution of data structures is considered part of the implementation process - this significantly reduces the amount of time spent on data modelling and mapping, and allows for a more agile and sustainable approach to development.

Approach summary

Through the implementation of these techniques, our goal is to streamline developer onboarding, resulting in a shorter time-to-development. Additionally, we aim to minimise the time and human effort needed for data architecture modifications, thus reducing the ongoing development impact. We also seek to decrease the level of modification required when incorporating new functionality into the platform. By eliminating the most common time sinks experienced during the development lifecycle we can allow developers to focus purely on the task at hand - create iteratively without fear of the classic impacts caused by refactoring for correctness, or to be more suitable for purpose.

6. Conclusion

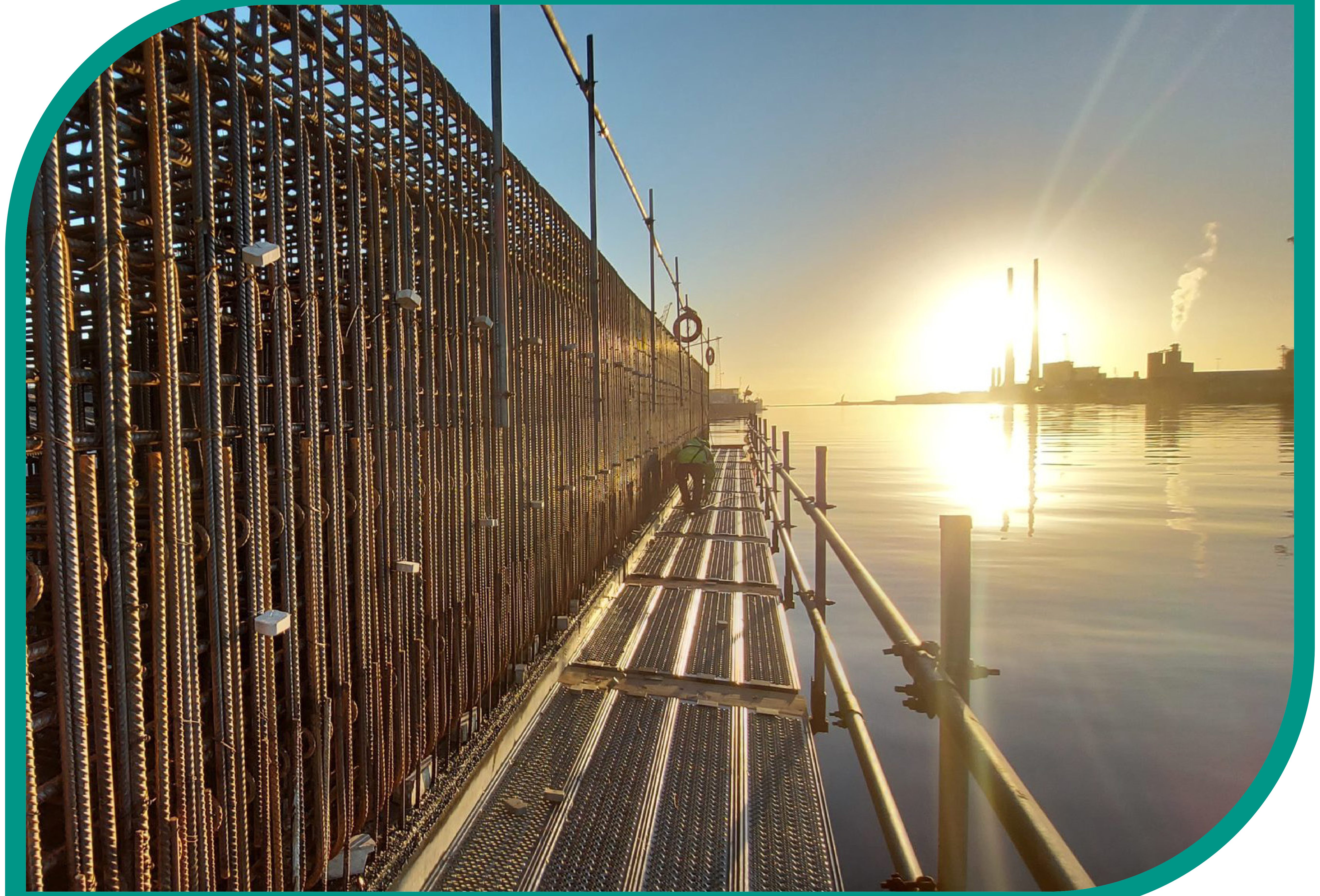
This white paper outlines a detailed approach to creating a serverless computing architecture with a focus on sustainability-first software development. We explore principles for designing carbon-aware software, emphasising energy-efficient coding, serverless architectures, and code optimisation for energy efficiency. While centred on Microsoft's Azure cloud platform, the theoretical framework applies to most cloud service providers. We believe that the technological shift into serverless computing is key to allowing manufacturers to reduce their environmental impact while maintaining operational efficiency and competitiveness in the market. We demonstrate this with a practical example of a serverless computing architecture, designed from a sustainability-first perspective, supported by data and industry research.

With multiple cloud service providers offering a diverse range of compute resources, it is essential that cloud solution architects are aware of sustainability practices and implement them accordingly in new developments and during migrations. There are many opportunities for the development of practical implementations and research to explore and contribute to this area of work.

Future work will focus on optimising steel production and minimising waste in rebar cutting, offering considerations from a technological viewpoint and the broader greener community. This includes ongoing evaluations of architectural designs to incorporate current developments and optimise computing resource allocation in serverless cloud environments for minimal waste and maximum utilisation. Our aim for the whitepaper is to serve as a guide for creating sustainable serverless computing architectures but also encourage ongoing efforts to adapt to evolving technologies and contribute to broader environmental sustainability goals across manufacturing.

7. Future work

Future work focuses on addressing critical aspects of minimising waste in rebar cutting, in part by operating within a more sustainable architecture design and optimising computing resource allocation in serverless cloud environments. There are several considerations from our viewpoint and the wider greener community at large. As cloud compute resource offerings are constantly evolving, we believe it is the architect's responsibility to be aware of the most current technologies which can support the Green Foundation's core tenets of sustainability. Architectural designs should be re-evaluated regularly to take current developments into account and redesign previous best practices accordingly. Another consideration is how best to allocate computing resources in serverless cloud environments to ensure minimal waste and maximum utilisation, to contribute to sustainability goals. We will apply this to our steel manufacturing use case and look to implement Carbon Aware SDK tools to optimise the current emissions outputs for data centres and regions for our technology solution. Of course, ensuring robust security measures within serverless cloud architectures to protect sensitive manufacturing data from breaches or unauthorised access without compromising sustainability requirements are also core. Using the REST API endpoints, we will query to support decision making around the optimal location software should be run. We will also investigate getting a carbon intensity for our application which will allows us to measure progress in reducing carbon emissions. This may also draw on the methodology currently in-development by the European Green Digital Coalition (EGDC), to help us assess the overall environmental impact of our digital solutions. Each of these factors contributes to advancing sustainability in manufacturing and broadens the scope of the environmental footprint from a technical viewpoint.



8. Work with us

Nightingale HQ - helping manufacturers digitalise

We specialise in helping manufacturers on their digitalisation journey. Our approach includes conducting Digitalisation Reviews to assess our clients' status and identify optimal ways to leverage technology for business improvement. We focus on optimising data, streamlining processes, and promoting sustainable manufacturing practices. We partner with a diverse range of manufacturers across Europe from steel to electronics. Our combination of expertise in data science, technology, and manufacturing, along with a strong customer-centric approach helps to accelerate digitalisation We are Microsoft partners, members of The European Institute of Innovation and Technology (EIT) in Manufacturing and Made Smarter in the UK. We are committed to driving innovation and excellence across the industry.

If you want to improve your manufacturing business or partner with us, please get in touch.

For more information please contact:

Ruth Kearney
CEO, Nightingale HQ
ruth@nightingalehq.ai

Chris Wilson
CTO, Nightingale HQ
chris.wilson@nightingalehq.ai

References

1 Amazon - What is Cloud Native?

2 AWS Customer Carbon Footprint Tool

3 AWS Well-Architected

4 Awesome Green Software

5 Azure Emissions Impact Dashboard

6 Boost Performance with Accelerated HPC and AI

7 Breakthrough Agenda

8 Breakthrough Agenda Report 2023

9 Carbon Aware SDK

10 Carbon Aware Tools

11 Carbon Aware Computing Whitepaper

12 Carbon Aware vs Carbon Efficient - Microsoft

13 Climate Champions - Steel Breakthrough: Priority International Actions For 2023

14 Cloud Carbon Footprint

15 Cloud Native Computing Foundation (CNCF) - Definition

16 Cutting Waste Minimization of Rebar for Sustainable Structural Work: A Systematic Literature Review

17 European Green Digital Coalition

18 Google - Cloud Architecture Framework

19 HPE - HPC Manufacturing

20 Apium - Green Software and Carbon Hack 2022

21 Green Software Foundation

22 Kaseya - High Availability

23 Deloitte - High performance computing in AI

24 How environmentally friendly organizations can use green coding to drive long-term success

20 HPC and AI

21 Weka - HPC Applications

22 IBM - Containerisation

22 IBM - High Performance Computing

23 IBM - What is a Digital Twin

24 Intel - HPC Artificial Intelligence

23 McKinsey & Company - Developer Velocity: How software excellence fuels business performance

24 Microsoft - Azure Cosmos DB Use Cases

24 Microsoft – Cloud Native

25 Microsoft - Developer Velocity

26 Microsoft - OpenAI Azure Supercomputer

28 Microsoft Well-Architected Framework

29 Microsoft - What is Data API builder?

27 Realising the potential of AI and HPC

28 Scientific Computing - HPC at the core of green and digital transition

29 Software Carbon Intensity

30 Shrink That Footprint - Carbon Footprint of Training GPT-3 and LLMs

31 Troy Hunt - Serverless to the Max: Doing Big Things for Small Dollars with Cloudflare Workers and Azure Functions

32 United Nationals Global Compact: Sustainability Report 2021

33 United Nations Environment Programme

34 IEC 62264-1:2013 Enterprise-control system integration. Part 1: Models and terminology

35 Industry 4.0 Core Information Model

36 CIMdata Releases eBook on Sustainability

Image references

Cover page:	Photo by Casey Horner on Unsplash
Cover page:	Midland Steel
Page 1:	World image by NASA on Unsplash
Page 4:	NHQ Generic Architecture
Page 7:	The Earth and I by Noah Buscher on Unsplash
Page 8:	ECMWF data centre, Bologna, Italy, September 2021 on flicker
Page 10:	Programmer by Christina @ wocintechchat.com on Unsplash
Page 11:	HPC Facility by Matt Howard on flicker
Page 13:	ECMWF's Atos HPC facility by Stefano Marzoli on Flicker
Page 14:	Cloud Image by rawpixel.com on Freepik
Page 17:	Dual power supply, ECMWF data centre by Stefano Marzoli on Flicker
Page 18:	ThisisEngineering RAEng on Unsplash
Page 19:	ThisisEngineering RAEng on Unsplash
Page 21:	ThisisEngineering RAEng on Unsplash
Page 22:	Midland Steel
Page 23:	Midland Steel
Page 24:	Midland Steel
Page 25:	Midland Steel
Page 26:	Midland Steel

NightingaleHQ

helping manufacturers digitalise

nightingalehq.ai

eit Manufacturing

Co-funded by the European Union

UKRI Innovate UK