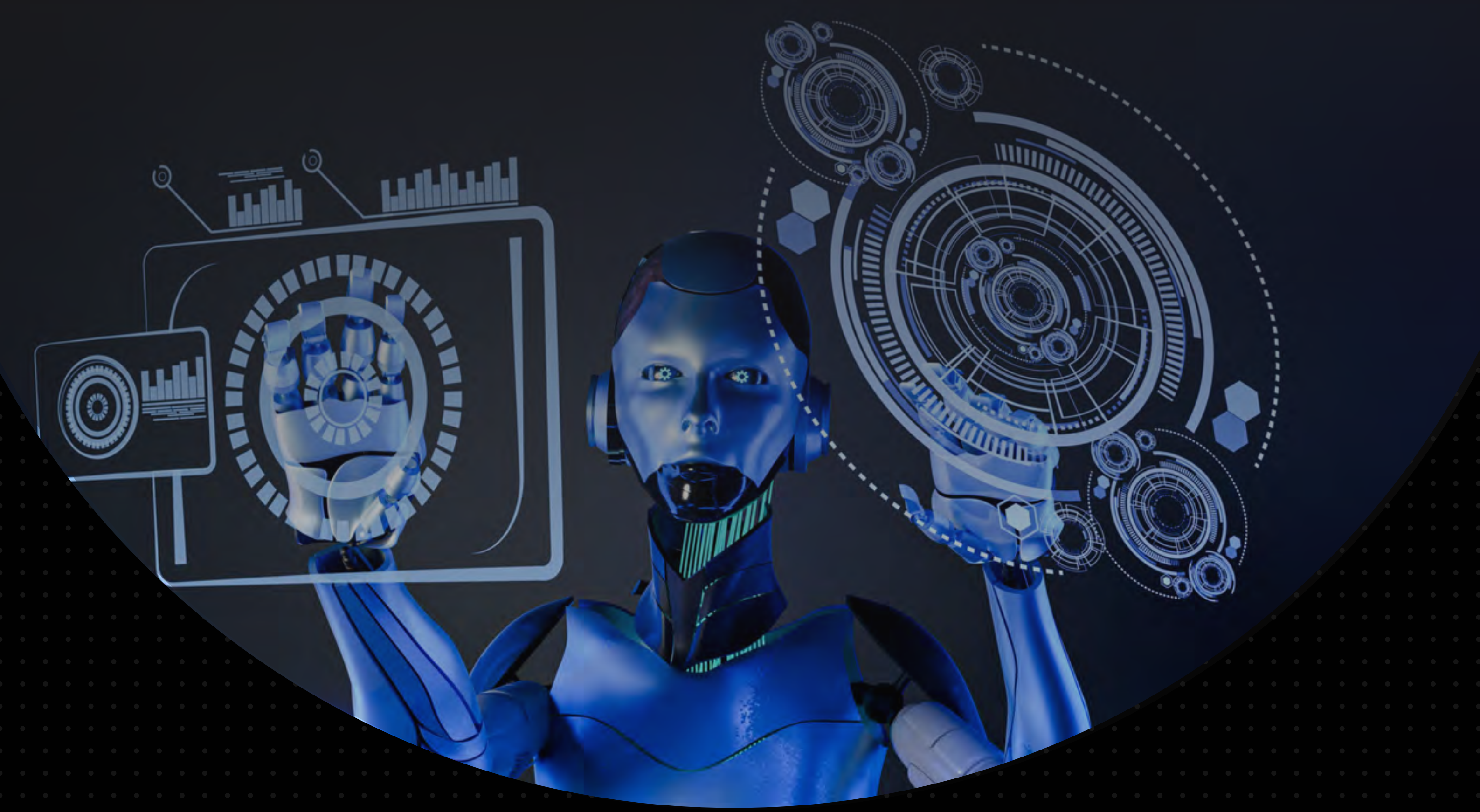
COREDGE

Co-Robot Automation Platform:

Streamlining Operational Efficiency

Introduction

CoRobots – A Cloud Native Workflow Automation Platform provided by Coredge.io is designed to enable collaborative development, managing dependencies and executing workflows while operating at scale, making it well suited for automations requiring complex workflows with multiple tasks.



Key Features

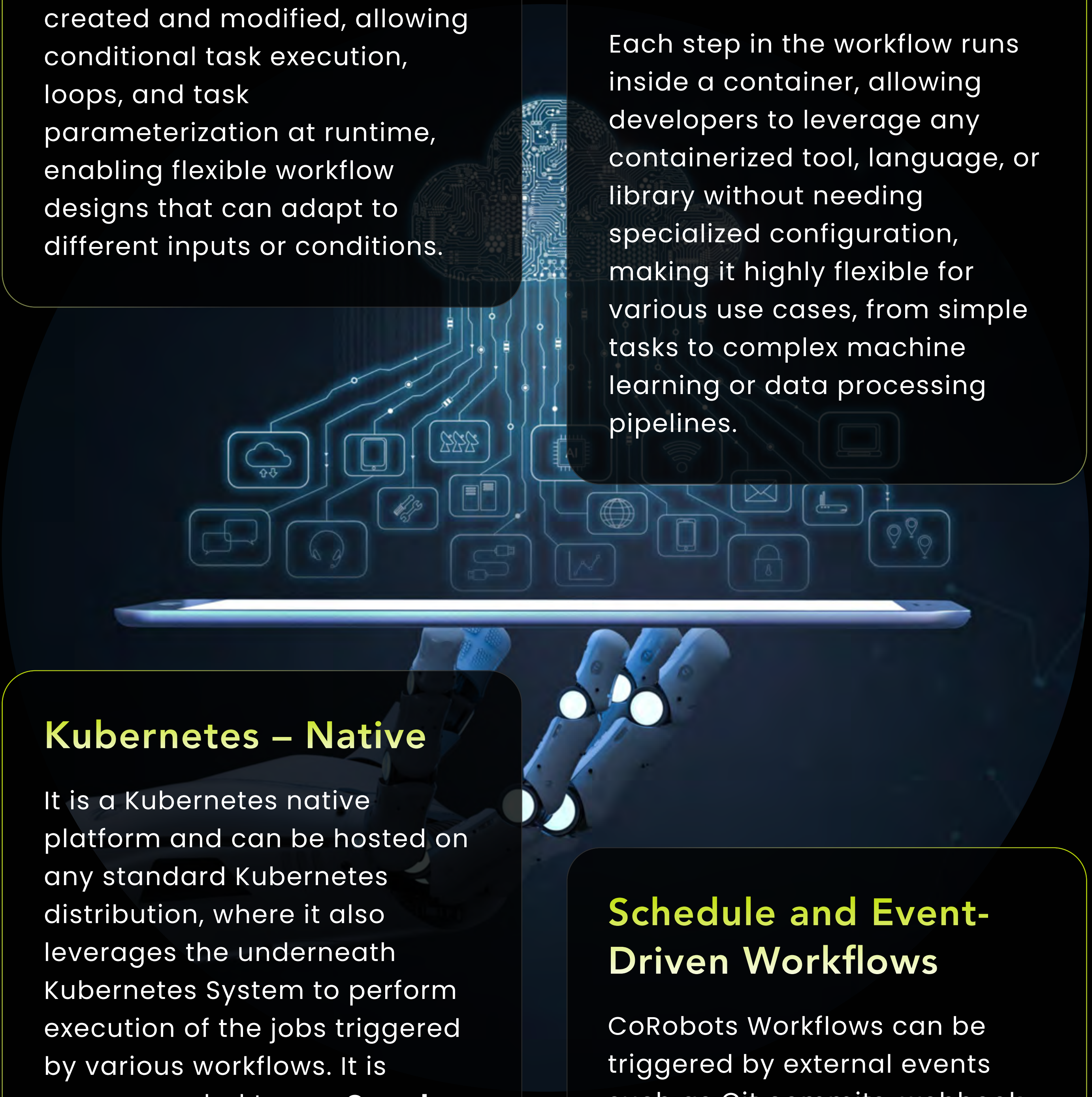
Following Key features makes CoRobots a powerful and flexible platform:

Multi Tenancy

Like all other Coredge provided platforms is inherently multi-tenant in nature providing separate workspaces for different teams, to develop and share their work

IAM/SSO

CoRobots provides options to integrate with existing Organization user repository supporting various mechanisms like OIDC, SAML, LDAP etc.



Dynamic Workflow Creation

Workflows can be dynamically created and modified, allowing conditional task execution, loops, and task parameterization at runtime, enabling flexible workflow designs that can adapt to different inputs or conditions.

Container Based Execution

Each step in the workflow runs inside a container, allowing developers to leverage any containerized tool, language, or library without needing specialized configuration, making it highly flexible for various use cases, from simple tasks to complex machine learning or data processing pipelines.

Kubernetes – Native

It is a Kubernetes native platform and can be hosted on any standard Kubernetes distribution, where it also leverages the underneath Kubernetes System to perform execution of the jobs triggered by various workflows. It is recommended to use **Coredge Kubernetes Platform (CKP)** to achieve better fine-tuned scalability of the platform

Schedule and Event-Driven Workflows

CoRobots Workflows can be triggered by external events such as Git commits, webhook events, or time-based schedules, making it suitable for CI/CD pipelines and automation.

Catalog

CoRobots allows different teams to develop individual tasks / jobs to workflow templates independently as part of different projects and then allow sharing it as part of common catalog with specific approvals.

Artifact Management

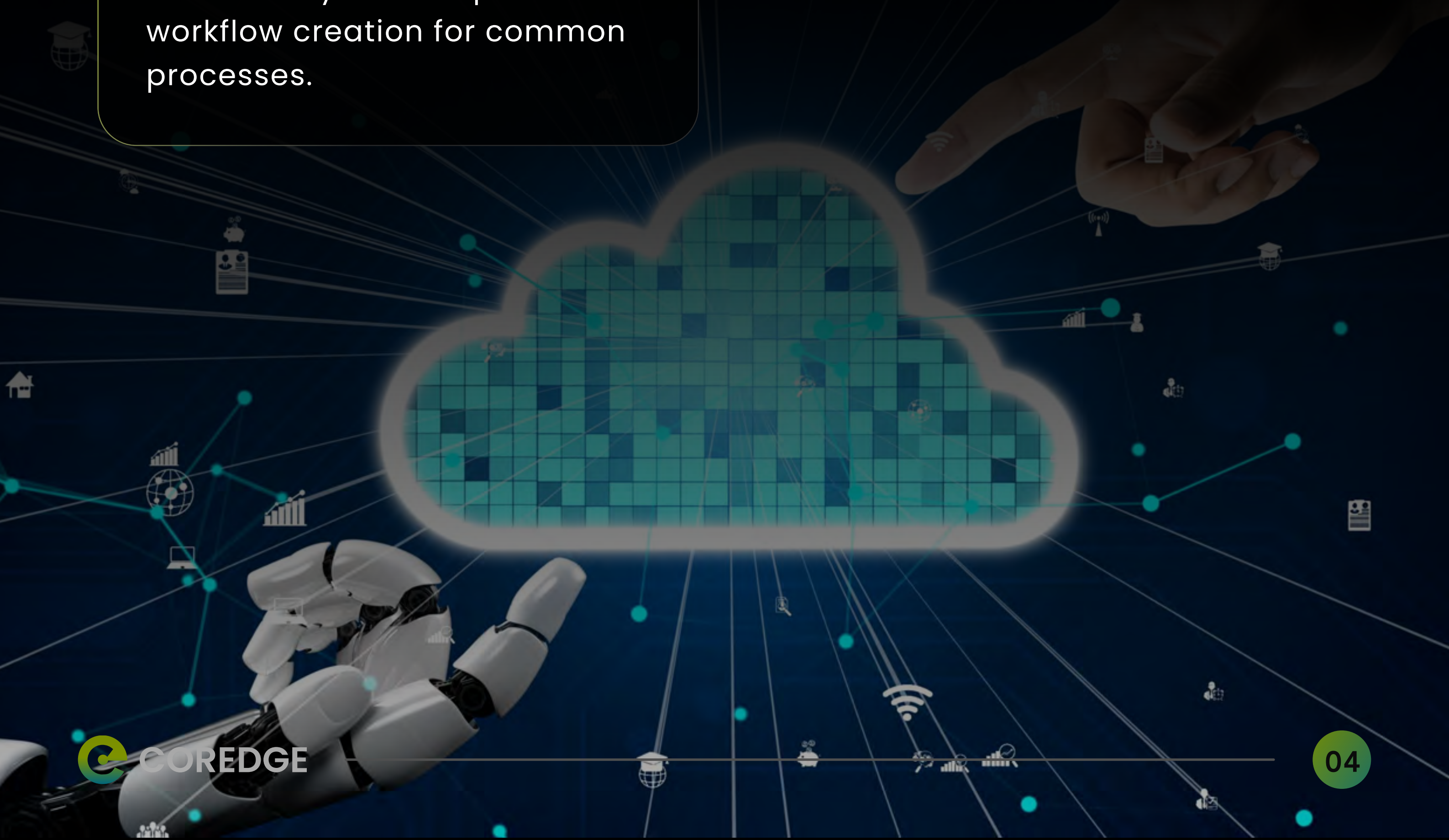
CoRobots while working with container-based execution still allow users to work with in browser editors or git repo references to internally build the relevant container images, these images are locally stored and managed.

Workflow Templates and Reusability

Users can define workflow templates for repeated use, promoting reusability and consistency across similar workflows, reducing redundancy and simplifies workflow creation for common processes.

Version Management

While new versions of the workflow templates and jobs are contributed it also maintains the version history, providing options to fallback to older versions for any specific requirements



Parallelism and Scalability

CoRobots supports highly parallel workflows, allowing large-scale execution of tasks across the Kubernetes cluster. It scales seamlessly by leveraging Kubernetes inherent ability to manage container workloads and resource allocation.

DAG (Directed Acyclic Graph) Support

Workflows are structured as Directed Acyclic Graphs (DAGs), enabling users to define task dependencies explicitly, while ensuring that tasks are executed in the correct order based on their dependencies, while allowing multiple tasks to run in parallel, ensuring efficiency.

Use Case No.01

Automated Device Monitoring and Maintenance in Data Centers

Problem: Monitoring and maintaining data centre infrastructure involves managing hundreds or thousands of devices, such as servers, switches, storage systems, and firewalls. Manually monitoring device health, performing updates, and addressing faults is time-consuming.

Use Case:

Event: When a device sends an alert (e.g., high CPU usage, memory failure, or hardware health issue) or regular health check data is received.

Workflow:

- The platform ingests the event (device health alert).
- A pre-built workflow template analyses the event to identify the type and severity of the issue.
- Modular workflow images trigger automatic remediation actions like restarting the device, applying patches, or reallocating workloads.
- If the issue is critical, escalate it by creating a service ticket and notifying relevant engineers.

***Outcome:** Automated device monitoring and maintenance ensure continuous operations and reduce downtime without manual intervention.*

Use Case No.02

Automated Resource Scaling for Critical Workloads in Cloud

Problem: Large scale end-user facing workloads are often resource-intensive and require dynamic scaling of compute, memory, and storage resources in cloud to meet varying demands.

Use Case:

Event: High CPU/GPU usage, memory consumption, or storage capacity thresholds reached during access of application or inferencing of AI models.

Workflow:

- The event (resource consumption threshold) triggers an automatic resource scaling workflow.
- Once the workload is reduced, the workflow scales down resources to optimize costs.
- The system continuously monitors usage metrics and sends notifications to administrators in case of resource bottlenecks.
- Workflow images are triggered to allocate additional compute resources (e.g., scaling virtual machines or containers in the cloud) based on demand.
- Remove the scaled down resources back to a Load Balancer of service discovery framework.
- Add the scaled up resources back to a Load Balancer of service discovery framework.

***Outcome:** Automated resource scaling ensures optimal use of cloud infrastructure, preventing performance bottlenecks during AI/ML training or inference workloads as well as large user facing applications.*

Use Case No.03

Data Preprocessing Automation for AI/ML Pipelines

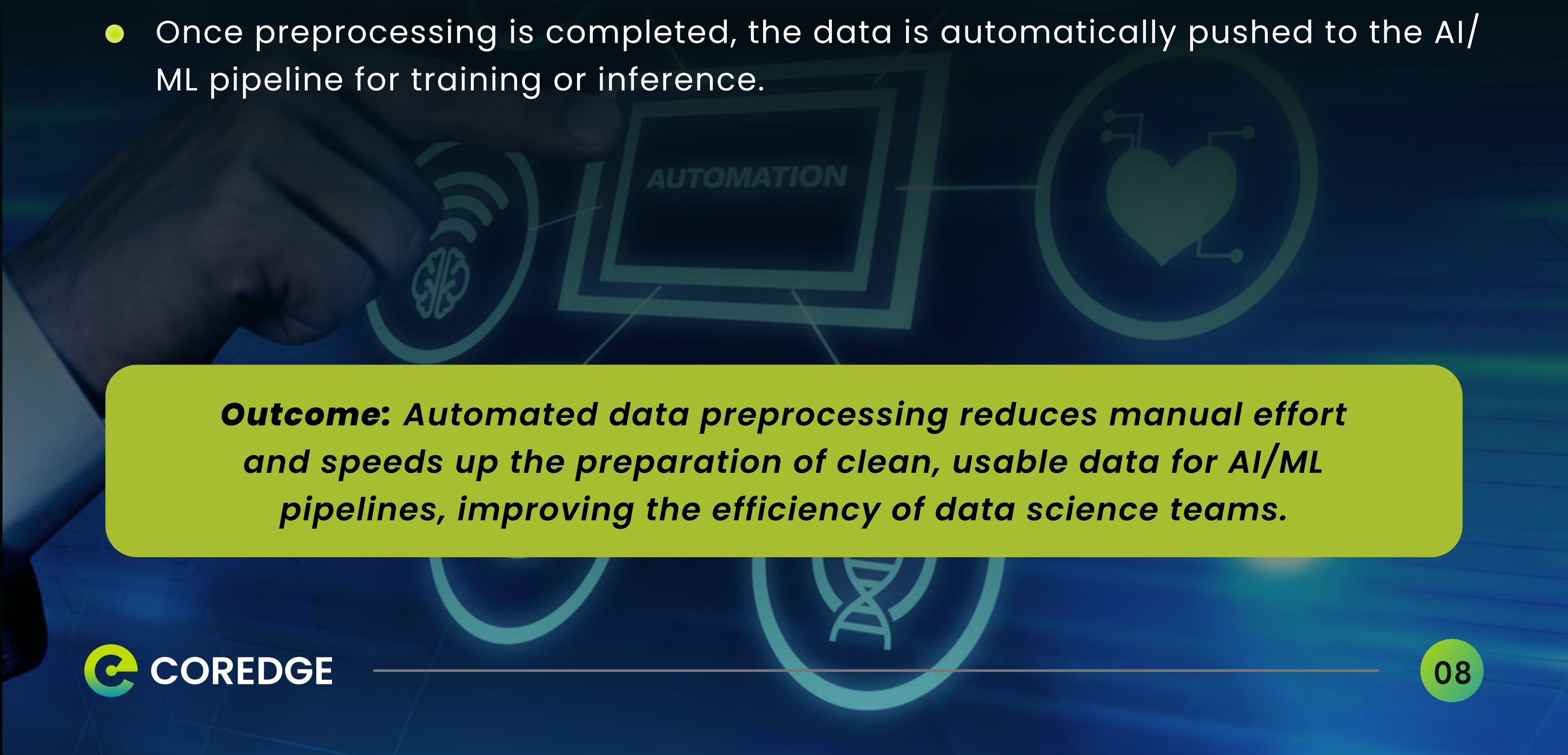
Problem: AI/ML workflows often require large-scale data preprocessing before feeding it into machine learning models. Automating the extraction, transformation, and loading (ETL) of data reduces time and effort in AI/ML data preparation.

Use Case:

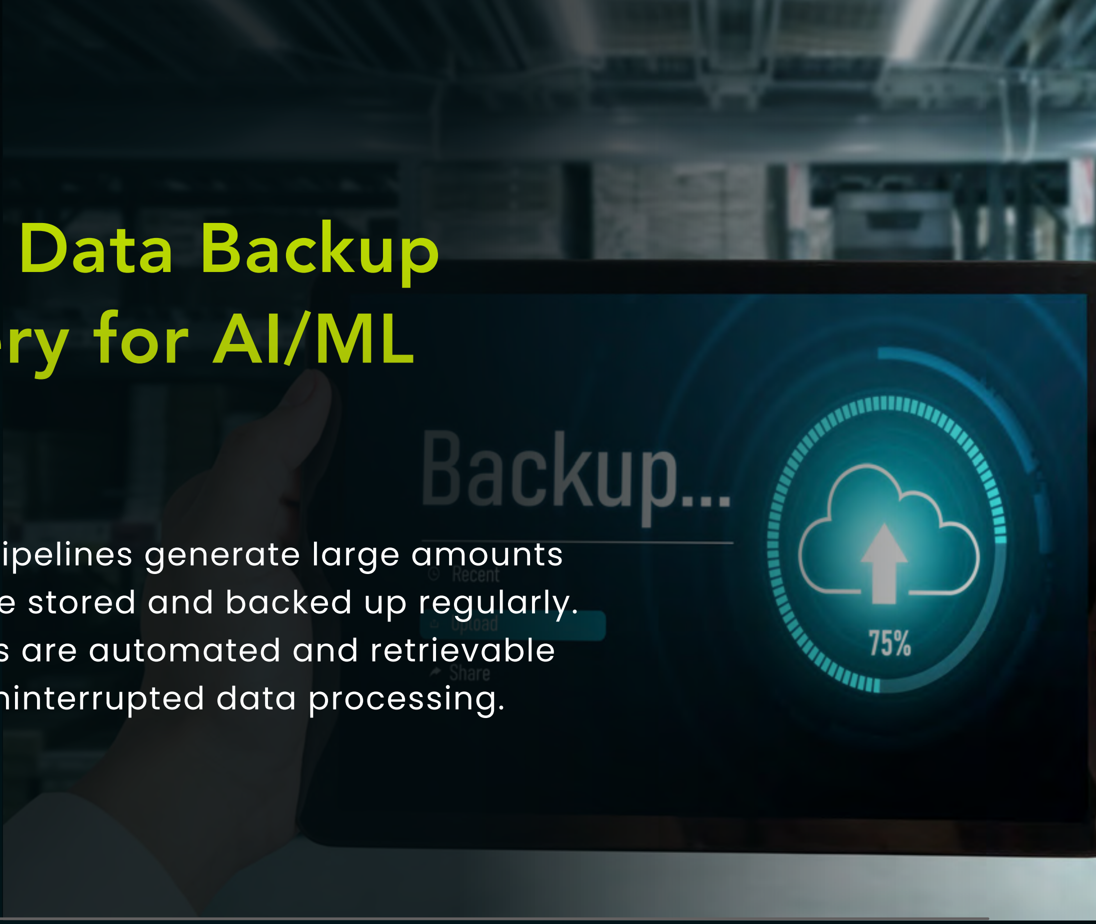
Event: When raw data is ingested from various sources (e.g., databases, APIs, IoT sensors).

Workflow:

- An event triggers the ETL workflow for data preprocessing.
- The platform's modular workflow images extract data, clean it (removing null values or errors), normalize it, and format it as required by the AI/ML pipeline.
- Workflow images can also apply feature engineering, such as encoding categorical variables or scaling numerical data, before passing it to the ML model.
- Once preprocessing is completed, the data is automatically pushed to the AI/ML pipeline for training or inference.



***Outcome:** Automated data preprocessing reduces manual effort and speeds up the preparation of clean, usable data for AI/ML pipelines, improving the efficiency of data science teams.*



Use Case No.04

Automated Data Backup and Recovery for AI/ML Pipelines

Problem: AI/ML data pipelines generate large amounts of data that need to be stored and backed up regularly. Ensuring data backups are automated and retrievable quickly is critical for uninterrupted data processing.

Use Case:

Event: When a new dataset is ingested or a machine learning job is completed.

Workflow:

- An event triggers the ETL workflow for data preprocessing.
- The platform triggers a workflow based on the event (dataset ingestion or job completion).
- A pre-built backup workflow template initializes an automated data backup process, storing the dataset on distributed storage systems.
- Workflow images manage compression, encryption, and verification of backups, ensuring data security.
- In the event of a failure or loss of data, the system triggers an automated data recovery workflow.

***Outcome:** Seamless backup and recovery of AI/ML datasets and models, ensuring high data availability and security in production environments.*

Use Case No.05

AI/ML Model Deployment and Monitoring

Problem: Deploying and monitoring machine learning models in production environments requires continuous tracking of model performance, accuracy, and operational health to ensure they function optimally.

Use Case:

Event: When new data is ingested or a machine learning model's performance metrics fall below a predefined threshold.

Workflow:

- The platform ingests the event (new data or performance degradation).
- A workflow triggers model re-evaluation and retraining using modular workflow images that integrate with data science tools (e.g., TensorFlow, PyTorch).
- The workflow deploys the updated model to production after validation and automatically updates monitoring dashboards.
- If the model fails to improve, the system alerts data scientists or suggests alternative model parameters.

***Outcome:** Continuous and automated monitoring and retraining of AI/ML models maintain high model accuracy and reduce manual efforts in managing model lifecycles.*

Use Case No.06

Oracle Database Provisioning Automation

Problem: Automatically provision Oracle databases in a self-service manner for internal teams or external customers.

Use Case:

Event: A user or system submits a request for a new Oracle database instance through a service portal or API.

Workflow:

- **Trigger Event:**

  The platform ingests the provisioning request (e.g., request for an Oracle 19c database instance).

- **Template Selection:**

  A pre-built workflow template for Oracle database provisioning is selected based on the request parameters (version, size, environment).

- **Validation and Approval:**

  Validate the user's credentials and check available BareMetal resources (e.g., storage, compute) in the cloud or data centre. If necessary, route the request to an approver using an approval workflow.

- **Provisioning Workflow:**

  The workflow image executes scripts to automatically deploy the Oracle database instance. Parameters such as database version, size, location (region or az), and performance settings (CPU, memory) are configured automatically.

- **Configuration and Security:**

  Apply security configurations (encryption, user roles, and access control) using modular workflow images. Configure Oracle Data Guard or Oracle RAC for high availability, if required.

- **Notification and Access:**

  Once provisioning is complete, notify the user with database connection details. Update the centralized system with the newly provisioned instance details for future management.

- **Post-Provisioning Actions:**

  Trigger automated backups and monitoring to ensure continuous operation.

***Outcome:** Oracle databases are provisioned automatically with minimal manual intervention, reducing the time from request to delivery and ensuring consistency in deployments.*

Use Case No.07

Automated Oracle Database Scaling

Problem: Automatically scale Oracle databases up or down based on performance metrics to ensure optimal resource utilization.

Use Case:

Event: Performance monitoring detects high CPU, memory usage, or storage limits on an Oracle database instance.

Workflow:

- **Trigger Event:**

  The system ingests real-time performance metrics (e.g., CPU usage exceeding 80%) and triggers the scaling workflow.

- **Resource Evaluation:**

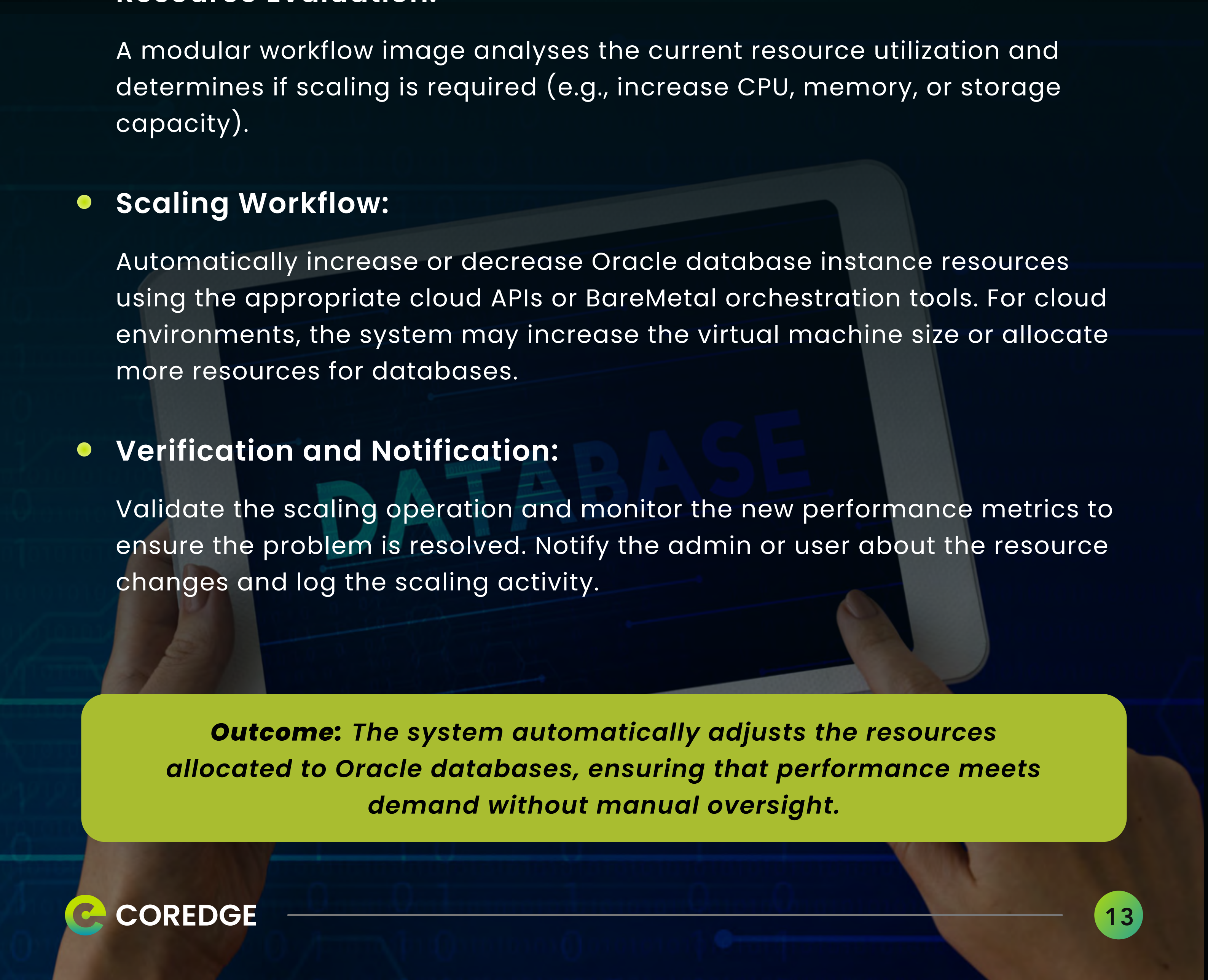
  A modular workflow image analyses the current resource utilization and determines if scaling is required (e.g., increase CPU, memory, or storage capacity).

- **Scaling Workflow:**

  Automatically increase or decrease Oracle database instance resources using the appropriate cloud APIs or BareMetal orchestration tools. For cloud environments, the system may increase the virtual machine size or allocate more resources for databases.

- **Verification and Notification:**

  Validate the scaling operation and monitor the new performance metrics to ensure the problem is resolved. Notify the admin or user about the resource changes and log the scaling activity.



***Outcome:** The system automatically adjusts the resources allocated to Oracle databases, ensuring that performance meets demand without manual oversight.*

Use Case No.08

Oracle Database Backup and Restore Automation

Problem: Automate Oracle database backup schedules and provide self-service restore options in case of failures or data corruption.

Use Case:

Event: Scheduled time for backup execution or a user request for backup or restore.

Workflow:

- **Backup Scheduling Trigger:**

  The platform ingests a scheduled event (e.g., daily, weekly backup). A modular workflow template for Oracle database backup is selected.

- **Pre-Backup Check:**

  Workflow checks database health and ensures all transactions are committed before starting the backup. It triggers Oracle Recovery Manager (RMAN) or cloud-native backup tools based on the environment.

- **Backup Execution:**

  The workflow template initiates the backup process, ensuring compression and encryption of data as per policy. Backup is stored in the desired location (on-premises storage or cloud).

- **Verification and Notification:**

  The workflow verifies the integrity of the backup. It logs the event, updates the backup repository, and sends a notification to administrators.

- **Restore Workflow (On Demand):**

  In case of a restore request, the platform triggers the restore workflow. The modular restore workflow retrieves the latest backup from storage and performs the restoration process while maintaining data consistency. Notifications are sent upon successful restoration.

***Outcome:** Automated and consistent backups of Oracle databases, with the ability to restore them quickly, ensuring minimal downtime in case of failures.*

Use Case No.09

Automated Provisioning of Load Balancer Instances

Problem: Enable self-service provisioning of load balancer instances based on user requests or application deployment needs.

Use Case:

Event: A user or an automated system requests the creation of a new load balancer instance (e.g., when deploying a new application).

Workflow:

- **Request Trigger:**

  The platform ingests a request (e.g., API or service portal request) to provision a new load balancer.

- **Template Selection:**

  The system selects a pre-built workflow template for load balancer provisioning, tailored to the user's needs (e.g., Layer 4 or Layer 7 load balancer, HTTP/HTTPS support).

- **Configuration and Resource Allocation:**

  The workflow configures the load balancer instance by defining parameters like target server groups, health check protocols, and traffic routing rules. Automatically allocate resources (IP addresses, bandwidth) to the load balancer based on the environment (on-premises or cloud).

- **Health Checks and Security Settings:**

  Configure health checks for backend servers and set up SSL termination or traffic encryption if needed. Apply firewall rules, DDoS protection, and access control lists (ACLs) to secure the load balancer.

- **Provisioning and Notification:**

  Deploy the load balancer instance and verify successful provisioning. Notify the user or system with the load balancer configuration details (e.g., IP addresses, DNS names).

- **Post-Provisioning Monitoring:**

  Automatically set up monitoring to track load balancer performance (e.g., traffic levels, response times, error rates).

***Outcome:** Self-service load balancer provisioning that ensures users can quickly spin up new instances without manual intervention, reducing time-to-market for applications.*

Use Case No.10

Dynamic Traffic Scaling Based on Network Load

Problem: Automatically scale the load balancer instance based on real-time network traffic and application demand to ensure optimal performance and cost-efficiency.

Use Case:

Event: An increase in network traffic or backend server load is detected through performance monitoring.

Workflow:

- **Trigger Event:**

  Performance metrics from the load balancer (e.g., high request rates, CPU/memory thresholds) trigger a scaling workflow.

- **Traffic and Load Analysis:**

  A modular workflow image analyses current traffic patterns, backend server performance, and application behaviour to determine scaling needs (e.g., increase/decrease the number of backend servers or load balancer instances).

- **Auto-Scaling Workflow:**

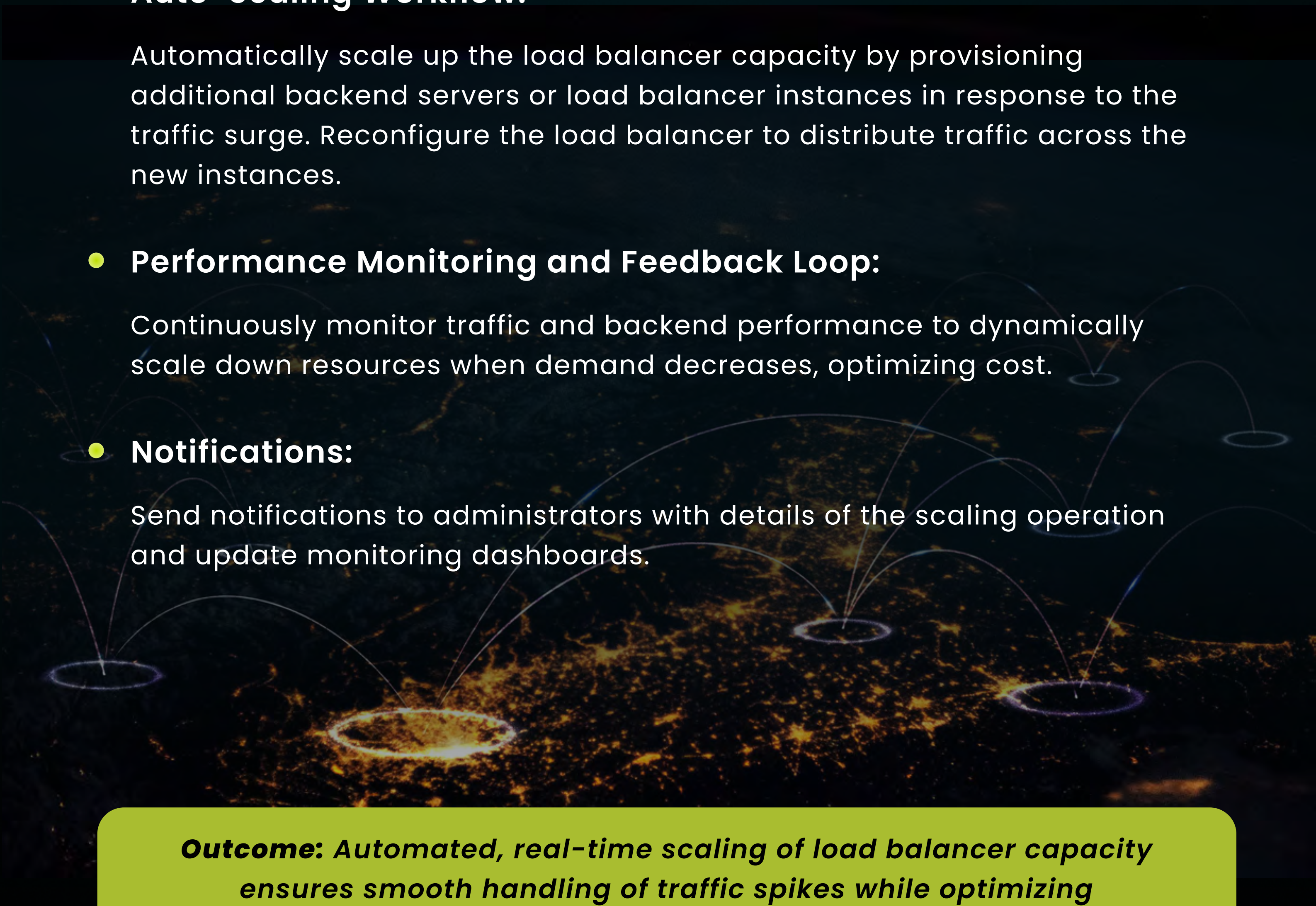
  Automatically scale up the load balancer capacity by provisioning additional backend servers or load balancer instances in response to the traffic surge. Reconfigure the load balancer to distribute traffic across the new instances.

- **Performance Monitoring and Feedback Loop:**

  Continuously monitor traffic and backend performance to dynamically scale down resources when demand decreases, optimizing cost.

- **Notifications:**

  Send notifications to administrators with details of the scaling operation and update monitoring dashboards.



***Outcome:** Automated, real-time scaling of load balancer capacity ensures smooth handling of traffic spikes while optimizing resource usage and reducing costs during low-traffic periods.*

COREDGE

About Coredge

Coredge is a leading provider of cloud-native and AI-driven solutions, leveraging a cutting-edge approach to deliver high-performing cloud infrastructure, Kubernetes, and hyper-converged technologies. With its Coredge Cloud Suite (CCS), the company enables businesses to build and scale multi-cloud environments while simplifying management through automation and integration. Coredge supports organizations in developing cloud-native applications, modernizing legacy systems, and optimizing infrastructure performance. As a trusted partner to major telco, SMEs, & Government, Coredge drives digital transformation, helping businesses innovate and thrive in a rapidly evolving technological landscape.

Noida

Office No. 2, 6th Floor,
Tower-B, Embassy Galaxy Business
Park, A-44 & 45, Sector 62, Noida,
Uttar Pradesh 201309

Bangalore

BHIVE Workspace,
No. 467/468 Shri Krishna Temple
Road, Indira Nagar 1st stage,
Bengaluru, Karnataka - 560038