Journal of Computer Science and Frontier Technologies
ISSN Print: 3104-4204
ISSN Online: 3104-4212
CD ISSN: 3104-4220
https://doi.org/10.63313/JCSFT.9004

Development Practice of a Distributed Campus Carbon Footprint System Based on the Coze Platform——Technical Implementation of Low-Carbon Behavior Incentive Function for Teenagers

Qianhan Wei
International Department, Jinling High School |Hexi Campus, Nanjing 210019, China
Email: 10151524@qq.com

How to cite this paper: Wei, Q. H. (2025). Development Practice of a Distributed Campus Carbon Footprint System Based on the Coze Platform——Technical Implementation of Low-Carbon Behavior Incentive Function for Teenagers. Journal of Computer Science and Frontier Technologies, 1(1), 38-46. ISSN Print: 3104-4204; ISSN Online: 3104-4212.
https://doi.org/10.63313/JCSFT.9004
Published: 2025-08-13

Abstract

As global climate change intensifies, cultivating low-carbon behavior among teenagers has become an important topic in the education field. However, many teenagers lack an intuitive understanding of the carbon emissions generated by their daily behaviors, and the effectiveness of traditional advocacy methods is limited. This research developed a distributed campus carbon footprint system based on the Coze low-code platform. Addressing the challenge of students struggling to quantify the carbon emissions of daily behaviors, the system realizes carbon footprint visualization and behavior guidance through modules such as behavior data collection, distributed point calculation, gamified carbon point incentives, and real-time suggestion pushing. Key technologies involved include user sharding storage strategy, dynamic carbon point algorithm, and transactional exchange engine. The core of the system lies in promoting the formation of students' low-carbon habits through instant feedback and positive incentives, providing a feasible technical solution for campus carbon neutrality.

Keywords

Distributed System; Coze Platform; Low-Carbon Education; Behavior Incentive; Campus Carbon Neutrality

Introduction

To address the challenges of global climate change, cultivating teenagers' awareness and practical capabilities regarding low-carbon environmental protection has become a crucial mission in current education. However, many teenagers lack intuitive cognition of the carbon emissions generated by daily behaviors such as commuting,

electricity usage, and paper consumption. Traditional classroom advocacy models suffer from insufficient interactivity and delayed feedback, making it difficult to effectively foster long-term green behavior habits. To tackle this challenge, this research designed and implemented a distributed campus carbon footprint system based on the Coze low-code platform. The system focuses on quantifying the carbon footprint of students' daily behaviors, transforming abstract environmental concepts into participable digital practices through innovative technical means and incentive mechanisms, thereby providing effective support for achieving campus carbon neutrality goals.

1. System Overall Architecture and Design Philosophy

The system adopts a distributed microservices architecture deployed on the Coze platform, fully leveraging its high efficiency in low-code development and elastic scaling capabilities. The data storage employs a "user sharding" strategy, distributing data across different database nodes based on students' classes or grades. The core design philosophy is to achieve "behavior digitalization, footprint visualization, and incentive instantiation". The system mainly consists of four functional modules:

1.1. Multi-source Behavior Data Collection Module

This module is responsible for collecting students' daily low-carbon behavior data, supporting multiple data input methods. Students can self-report behaviors such as walking distance, paper saving, turning off appliances, etc.

1.2. Distributed Carbon Point Calculation Engine

This module is the core module of the system, responsible for quantifying the carbon reduction amount of behaviors in real-time and calculating corresponding points. It adopts a calculation model of "base value × dynamic coefficient". For example: Walking 1 kilometer = base points 5 points, rainy day coefficient increases to 1.5 times.

1.3. Gamified Incentive and Exchange Module

This module provides a tiered reward system: Carbon points can be converted into redeemable rewards such as stationery usage rights, book coupons, and gamified elements like "Environmental Protection Guardian" badges; it also publishes class/grade carbon point leaderboards.

1.4. Real-time Low-Carbon Suggestion Push Module

Based on student behavior data analysis, it pushes personalized low-carbon action suggestions, such as "The temperature is suitable today, walking to school is recommended and can earn an extra 10% points."

2. Distributed Architecture Design and Key Technology Implementation

This system adopts a decentralized microservices architecture deployed on the Coze platform, with the core goal of addressing the challenges of high-concurrency access and massive user data in campus scenarios, while ensuring system scalability and high reliability. The architecture design strictly adheres to distributed system principles and mainly includes the following key layers and technology selections.

2.1. Service Layering and Communication Mechanism

The system uses Coze's built-in high-performance API gateway as a unified entry point, responsible for request routing, load balancing, JWT token-based authentication, and rate limiting/circuit breaking. The gateway automatically distributes requests to corresponding microservice instances. For example: Requests to the path /api/behavior are routed to the behavior-collect-service service using round-robin load balancing; requests to the path /api/points are routed to the carbon-calc-service service using least-connections load balancing.

2.2. Core Microservices and Data Storage

The business microservice layer is split into five independently deployed core services: User Service, managing student information and shard mapping; Behavior Collection Service, receiving and preprocessing sensor and user-reported data; Carbon Point Calculation Service, implementing the core carbon point algorithm; Incentive Exchange Service, handling point consumption and gamified incentives; Push Service, generating and sending personalized low-carbon suggestions.

Each service has its own independent database and achieves dynamic management of service instances through the Coze platform's service registration and discovery center. The data storage layer adopts a database and table sharding strategy. Based on student ID, it uses consistent hashing to shard user data horizontally across multiple physical MySQL database instances. For example, for a school with 2000 students, the designed hash function can evenly distribute user data across 4 database instances, effectively dispersing read/write pressure. The core logic of the hash function is: shard_index = (studentId.hashCode() & Integer.MAX_VALUE)% shard Count.

2.3. Asynchronous Communication and Decoupling

Decoupling dependencies between microservices is achieved using message queues (e.g., RabbitMQ) for asynchronous communication. After validating the data, the Behavior Collection Service encapsulates it into a structured message containing fields like timestamp, student ID, behavior type, and behavior value, and publishes it to a designated message queue. The Carbon Point Calculation Service subscribes to this queue as a consumer, enabling non-blocking asynchronous processing of behaviors. The Coze platform simplifies the process of creating, binding, and monitoring mes-

sage queues.

2.4. Implementation of User Sharding Storage

The implementation of the user sharding storage strategy involves three key steps:

- Shard Key Design: Selecting the unique and frequently queried student_id as the primary shard key.
- Routing Logic: Maintaining a global shard mapping relationship in the User Service, stored in a Redis cluster. The data structure is key-value pairs, e.g., "S2025001": "db_shard_1". The API gateway parses the JWT token in the request to obtain the student_id, calls the User Service interface to locate the target database shard.
- Data Access Layer Encapsulation: Implementing an intelligent data access component within services that need to access user data. This component queries the shard mapping based on the student_id, automatically selects the corresponding database connection pool to perform operations, and is transparent to the upper-layer business logic. Coze platform's dynamic data source configuration functionality effectively supports the management of multiple shard data sources.

2.5. Cross-Shard Processing and Fault Tolerance Mechanism

For cross-shard queries (e.g., school-wide leaderboards), the system adopts a "push-merge" strategy. Each database shard calculates its local ranking data in parallel, and the Incentive Exchange Service aggregates these intermediate results and performs global sorting. Student data migration (e.g., changing classes) ensures transactional synchronization between shards through carefully designed background tasks. The system's elasticity and fault tolerance mechanisms cover the following three aspects:

- Service Instance Elastic Scaling: The Coze platform automatically scales microservice instances in or out based on preset CPU/memory thresholds and request queue length.
- Storage Layer High Availability: Each database shard is configured with master-slave replication for read-write separation. Coze's integrated database proxy component enables automatic failover.
- Distributed Caching: Frequently accessed data (e.g., user points, popular item inventory) is cached in a Redis cluster. A cache-first strategy is used: read requests query the cache first, fetching from the database and backfilling the cache on a miss; write requests update the database and then synchronously invalidate or update the cache; reasonable cache expiration times are set to prevent stale data.

3. Deep Implementation of Core Functional Modules: Carbon Point Algorithm and Calculation Engine

The Dynamic Carbon Point Algorithm model (DCIA) is the core of the system, aiming to scientifically quantify behavior value and achieve dynamic incentives.

3.1. Dynamic Carbon Point Algorithm Model (DCIA)

This model contains the following key elements:

3.1.1. Base Carbon Emission Saving (Base Saving)

Based on carbon emission factors published by authoritative sources like IPCC and China's Ministry of Ecology and Environment, calibrated with localized campus surveys. The system maintains a behavior baseline value mapping table in the Coze configuration center, e.g., walking 1 km saves 230 grams CO2e (Carbon Dioxide Equivalent), using electronic documents for 1 hour saves 50 grams CO2e, supporting dynamic updates.

3.1.2. Dynamic Adjustment Coefficient (Dynamic Factor)

Composed of the product of the Behavior Difficulty Coefficient (Difficulty Coeff) and the Environmental Contribution Factor (Environmental Impact Factor). The Behavior Difficulty Coefficient is dynamically adjusted based on environmental or behavioral attributes, e.g., walking on a rainy day coefficient increases to 1.5, coefficient for continuous waste sorting for 7 days or more is 1.2 + (continuous days // 7) * 0.1. The Environmental Contribution Factor is set based on the behavior's potential contribution to key campus environmental indicators, e.g., turning off appliances during peak electricity hours (e.g., 18:00-20:00) can have a factor of up to 1.3.

3.1.3. Normalization Factor (Normalization Factor)

Used to balance the point value across different behaviors. By analyzing the base saving range (Base Saving Range), expected frequency of occurrence (Frequency), and preset weight for implementation difficulty (Weight) of each behavior, combined with the system's target for students' expected daily points (Global Target Points Per Day, GTPD) and the estimated total daily carbon saving for the whole school (Estimated Total Saving Per Day, ETSDP), it is calculated by the formula: Normalization Factor = (GTPD / ETSDP) * (Base Saving * Frequency) * Weight.

3.1.4. Points Calculation Formula

The final carbon points calculation formula is: Carbon Points = Base Saving(behavior, parameters) * Dynamic Factor(behavior, user, context) * Normalization Factor(behavior). All model parameters and rules are stored in the Coze configuration center, supporting hot updates.

3.2. Implementation of Distributed Calculation Engine

The distributed carbon point calculation engine is responsible for executing the DCIA

algorithm efficiently, accurately, and consistently, processing massive asynchronous behavior messages.

3.2.1. Task Distribution and Parallel Processing

After behavior data messages arrive at the queue, the Carbon Point Calculation Service starts a multi-consumer thread group. The Coze task scheduler ensures message processing uniqueness. Consumers route tasks to the user's corresponding calculation shard based on the student_id in the message and the shard mapping, guaranteeing sequential updates for the same user's points. Each calculation shard has a dedicated thread pool for parallel task processing.

3.2.2. Computational Atomicity and Consistency Guarantee

The core calculation process includes: obtaining behavior configuration, querying user and environmental context, calculating the dynamic coefficient, applying the normalization factor to derive points. The key step is atomically updating user points, achieved through database transactions: Begin transaction → Update user total points using an SQL statement with row lock UPDATE user_points SET points = points + ? WHERE student_id = ? → Insert detailed behavior record → Commit transaction. If execution fails, the transaction is rolled back, and exceptions are handled via Coze platform's retry and dead-letter queue mechanisms.

3.2.3. Performance Optimization Strategies

Performance optimization strategies mainly include: Batching: Accumulating behavior data with lower timeliness requirements and processing/updating the database in batches. Local Caching: Frequently accessed configuration data and user static information are cached in the computing node's local memory (e.g., Caffeine). Asynchronous Logging: Detailed behavior records are written asynchronously to reduce pressure on core transactions.

4. Strong Consistency Guarantee of Transactional Exchange Engine

The core challenge of the incentive exchange module lies in guaranteeing absolute accuracy of inventory and user points under high concurrency scenarios, preventing overselling or point overdrafts. The engine adopts a combined solution of distributed transactions and multi-level locking mechanisms.

4.1. Core Exchange Process

4.1.1. Pre-check

After a user initiates an exchange request, the server quickly queries the user's point balance and item inventory in the cache. This requires only eventual consistency at this stage, solely for quickly filtering out obviously invalid requests.

4.1.2. Distributed Transaction Processing

After passing the pre-check, it enters the strong consistency transaction flow.

- Acquire Distributed Lock: Attempt to acquire a distributed lock based on the target item ID (e.g., Redis RedLock).
- Query and Validate with Row Lock: Within the database transaction, use SELECT ... FOR UPDATE to lock the user points record and the item inventory record for secondary strong consistency validation (sufficient inventory and points?).
- Update Inventory Optimistically: Execute inventory deduction using optimistic locking (version number or CAS): UPDATE inventory SET stock = stock - ?, version = version + 1 WHERE item_id = ? AND version = ?. If the number of updated rows is 0 (version conflict), rollback the transaction.
- Deduct Points and Create Order: Execute the SQL to deduct user points and create an exchange order record.
- Commit Transaction and Release Lock: If all operations succeed, commit the transaction and release the distributed lock. Failure at any step causes transaction rollback and lock release.

4.2. Multi-layered Consistency Guarantee Mechanisms

This implementation integrates four key mechanisms to ensure strong consistency:

- Distributed Lock (Redis RedLock): Controls global concurrent access at the item level.
- Database Row-Level Lock (SELECT ... FOR UPDATE): Locks data rows within a transaction to prevent concurrent read-write conflicts.
- Optimistic Lock (Version Check/CAS): Performs concurrency control during inventory updates to prevent overselling.
- Declarative Transaction Management (e.g., @Transactional): Guarantees atomicity (ACID) of database operations.

This multi-layered guarantee mechanism successfully achieved zero overselling and accurate point deduction under instantaneous high concurrency (e.g., during the pilot school's "Eco-stationery Gift Pack" event, 1200 items were exchanged within 3 seconds). The error handling mechanism includes automatic retries and dead-letter queue management; unsuccessful orders enter a manual review process. The Coze platform's support for distributed transactions provided a solid foundation for this.

5. Application Effect and Educational Value

This system can be deployed and operated in primary and secondary schools, achieving educational value in the following aspects.

5.1. Enhancing Carbon Footprint Awareness

Through their personal carbon account homepage, students can intuitively view the

carbon savings and accumulated points generated by their daily and weekly behaviors. The carbon footprint transforms from a vague concept into clear data metrics, significantly enhancing students' awareness of the environmental impact of their own behaviors.

5.2. Stimulating Participation Enthusiasm

Gamification mechanisms, such as point rankings, badge collection, and exchanging for desired items or benefits (e.g., sports equipment usage rights, book coupons, eco-stationery), can effectively stimulate students' enthusiasm. Environmental behaviors are transformed into enjoyable competitions and rewards, prompting students to actively seek and practice low-carbon opportunities. Habits like walking/cycling to school, paperless learning, consciously turning off lights, and the "Clean Plate" action can be reinforced.

5.3. Promoting Behavior Habit Formation

Every low-carbon choice can quickly gain system recognition, such as point increases and achievement unlocks. This continuous positive reinforcement effectively promotes the formation of sustainable behavioral habits, internalizing low-carbon awareness into daily actions.

5.4. Empowering Campus Education Management

This system can provide a powerful digital tool for the school's moral and environmental education, helping to achieve the campus's overall energy-saving and consumption-reduction goals, becoming a practical platform for promoting campus carbon neutrality.

6. Conclusion

The distributed campus carbon footprint system built on the Coze low-code platform, through its innovative technical architecture—including user sharding storage, the Dynamic Carbon Point Algorithm (DCIA), a distributed real-time calculation engine, gamified incentive mechanisms, and strongly consistent exchange transaction processing—helps solve the core problems of difficulty in quantifying, weak perception of, and lack of incentives for teenagers' low-carbon behaviors. This system transforms intangible carbon emissions into visualized point metrics and converts environmental advocacy into tangible instant feedback and positive incentives, enhancing teenagers' enthusiasm and persistence in participating in low-carbon practices.

Leveraging digital tools to transform abstract environmental responsibilities into concrete, quantifiable, and attractive daily actions is an effective path for cultivating sustainable behavioral habits in teenagers. This system not only provides an implementable technical solution for carbon neutrality goals in campus settings, but its core logic of "quantification-feedback-incentive" also offers a valuable reference

paradigm for the broader field of behavior cultivation education. Future work will focus on continuous optimization and localized calibration of the carbon point algorithm, deeper integration with smart campus IoT systems and educational administration systems, and tracking research and effectiveness evaluation of long-term behavioral changes, aiming to promote the application of this system on a larger scale and contribute to the enhancement of teenagers' ecological civilization literacy.

References

[1] You, M.Q. and Li, Sh. M. (2023) The Rule of Law Response to the "Double Carbon" Transformation and the Synergy of ModernEnvironmental Governance System from the Perspective of Meta-governance. Henan Social Sciences, 31,56-66.

[2] Phalake S V ,Joshi D S. and Rade A K , et al. (2023) Optimization for achieving sustainability in low code development platform. International Journal on Interactive Design and Manufacturing (IJIDeM), 18,5717-5724.

[3] Wu, H.Y. and Deng, W.J. (2020) Research Progress on the Development of Microservices. Journal of Computer Research and Development, 57,525-541.

[4] Zheng, M. R., Li, J. L., Wang N. F., Lin G. W., Wen, Q. M. and Zhong, J. H. (2024) Research and Implementation of Low-carbon Park Load Aggregator System Based on Low Code Cloud Platform. Automation System Engineering, 43,42-47.

[5] Xu, L., Qu, G.H., Li, G.R.Q., et al. (2024) Carbon Footprint Status and Carbon Neutral Willingness of College Students in Beijing, China. Journal of Catastrophology, 39,16-23.

[6] Yu, X., Sun, L.S., Sun M.N., and Fang, J.T. (2025) Innovative Research on Low-code Development Platforms for Enhancing User Experience and Personalized Customization Capabilities. Technology Innovation and Application, 15,13-16.