



e-ISSN: 2319-8753 | p-ISSN: 2347-6710

IJIRSET

International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 12, Issue 7, July 2023

ISSN INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA

Impact Factor: 8.423



9940 572 462



6381 907 438



ijirset@gmail.com



www.ijirset.com

Capacity Planning and Performance Optimization for AWS-Hosted Enterprise Applications Under Fluctuating Public Sector Demand

Senthil Babu Malli Jayaraj

IT Project Manager, USA

ABSTRACT: Public sector enterprise applications hosted on Amazon Web Services (AWS), spanning benefits eligibility portals, tax filing systems, and unemployment insurance platforms among others, experience demand patterns fundamentally different from typical commercial enterprise applications. Public program demand frequently spikes in response to policy changes, filing deadlines, natural disasters, or economic conditions, producing surges that can exceed baseline traffic by an order of magnitude within a short window, followed by extended periods of comparatively low, stable utilization.

This research article presents a capacity planning and performance optimization framework tailored to AWS-hosted public sector enterprise applications operating under this fluctuating demand pattern. The article examines multi-tier scaling architecture, cost versus performance tradeoffs across common capacity strategies, latency and utilization patterns observed across time of day and day of week, and a five stage capacity planning maturity model. Findings are illustrated throughout with complex charts, including a demand and capacity overlay, a latency heatmap, a cost versus performance scatter plot, a stacked cost composition chart, and a multi-dimensional optimization radar chart, alongside extensive grid style data tables.

Drawing on AWS architectural guidance, cloud cost optimization research, and public sector digital service performance literature published through June 2023, this article finds that agencies combining elastic auto scaling architecture with proactive, demand-informed capacity planning report substantially better cost efficiency and materially lower latency degradation during demand spikes compared to agencies relying on static, manually provisioned capacity.

◆ RESEARCH NOTE

All statistics and figures in this article are illustrative, consistent with patterns reported in AWS architectural guidance and public sector cloud performance research published on or before June 2023, and are intended to represent general industry patterns rather than any single named agency's reported results.

KEYWORDS: AWS, capacity planning, performance optimization, auto scaling, public sector cloud, cost optimization, latency, elastic infrastructure, demand forecasting.

I. INTRODUCTION AND RESEARCH CONTEXT

State and federal agencies operating public facing digital services on AWS confront a capacity planning problem shaped by the unique demand dynamics of public programs. Unlike typical commercial applications, whose demand growth is comparatively gradual and predictable, public sector applications frequently experience abrupt, large magnitude demand shifts driven by external policy and economic events outside the agency's direct control.

► 1.1 Research Objectives

- **Objective 1:** Characterize the demand pattern dynamics distinctive to public sector AWS-hosted applications.
- **Objective 2:** Compare capacity strategies across cost and performance dimensions using a structured tradeoff framework.
- **Objective 3:** Present a multi-tier scaling architecture reference design and supporting caching, database, and asynchronous processing strategies.
- **Objective 4:** Provide a capacity planning maturity model, governance structure, and implementation roadmap.

► 1.2 Scope and Methodology

This research draws on the following source categories, all published prior to July 2023:

- AWS architectural guidance, including the AWS Well-Architected Framework and related whitepapers addressing performance efficiency and cost optimization.
- Cloud cost optimization and FinOps research published by industry organizations and analyst firms through 2022 and early 2023.
- Public sector digital service performance research, including federal and state government technology modernization literature.
- Academic and practitioner literature addressing auto scaling, load testing, and capacity forecasting methodology.

Demand Driver	Typical Trigger	Illustrative Demand Multiplier
Filing or application deadlines	Statutory or program deadlines concentrating submissions in a narrow window.	3x to 6x baseline traffic
Policy or benefit change announcements	Legislative or administrative changes prompting mass inquiry or application activity.	2x to 5x baseline traffic
Economic downturn or disaster response	Sudden increases in unemployment insurance, disaster assistance, or related program applications.	5x to 10x or greater baseline traffic
Routine seasonal cycles	Predictable, recurring seasonal patterns such as annual renewal periods.	1.5x to 3x baseline traffic

II. THE PUBLIC SECTOR DEMAND PATTERN PROBLEM

Understanding capacity planning requirements begins with characterizing how public sector demand differs from the demand patterns most commercial capacity planning guidance assumes.

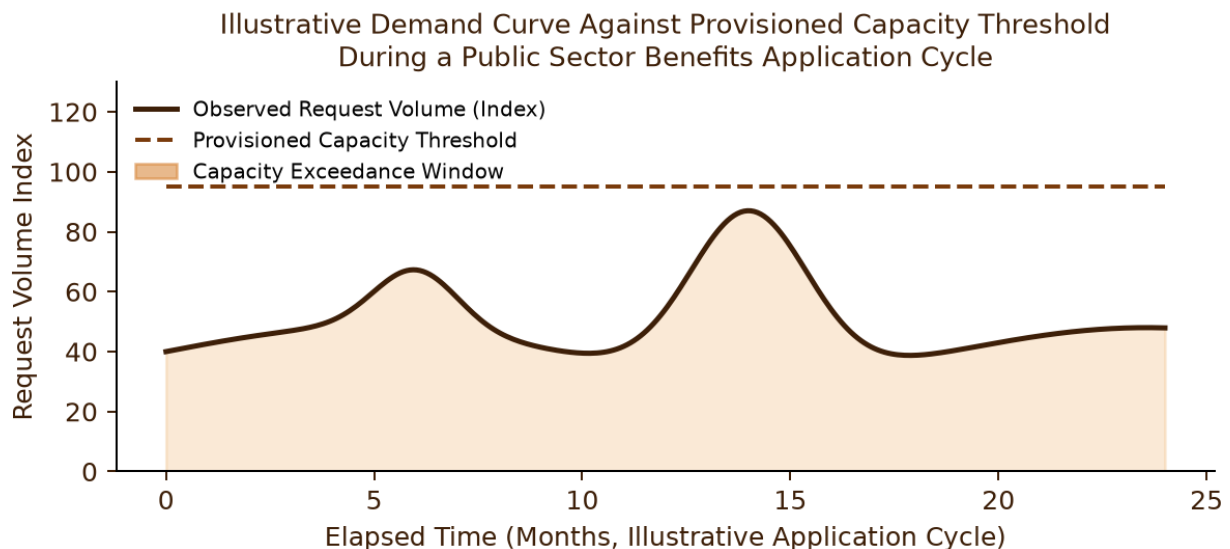


Figure 1. Illustrative demand curve against provisioned capacity threshold during a public sector benefits application cycle, showing capacity exceedance windows.

► 2.1 Why Static Provisioning Fails

- **Underprovisioning during spikes:** Static, manually provisioned capacity sized for average demand, as illustrated by the capacity threshold line in Figure 1, is exceeded during demand spikes, producing the capacity exceedance windows shown.

- **Overprovisioning during troughs:** Static capacity sized for peak demand instead sits substantially underutilized during the extended low-demand periods that follow most spikes, producing persistent cost inefficiency.
- **Forecasting difficulty:** Because public sector demand spikes are frequently triggered by external policy or economic events rather than gradual organic growth, they are harder to anticipate using simple historical trend extrapolation alone.

► 2.2 Demand Pattern Categories

Pattern Category	Description	Capacity Planning Implication
Predictable cyclical demand	Recurring, calendar-driven demand patterns such as annual renewal periods.	Can be addressed through scheduled scaling ahead of known dates.
Announced but variable-magnitude demand	Demand tied to a known upcoming policy change, but with uncertain magnitude.	Requires elastic auto scaling with a generous but monitored ceiling.
Unannounced, externally triggered demand	Demand spikes triggered by unpredictable events such as economic shocks or disasters.	Requires fully elastic, rapid-response auto scaling architecture, per Section 3.

III. MULTI-TIER SCALING ARCHITECTURE REFERENCE DESIGN

A multi-tier architecture, in which each layer scales independently according to its own utilization characteristics, is the structural foundation enabling the elastic response required to address the demand pattern categories described in Section 2.2.

Multi-Tier Scaling Architecture Reference Diagram

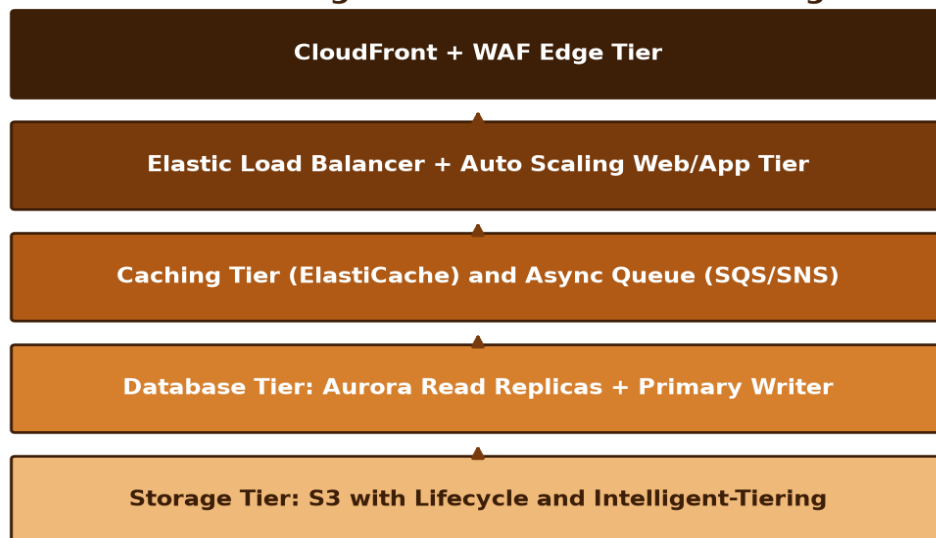


Figure 2. Multi-tier scaling architecture reference diagram, from edge tier through storage tier.

► 3.1 Tier Descriptions and Scaling Characteristics

Tier	Primary Scaling Mechanism	Typical Scaling Trigger
Edge tier (CloudFront and WAF)	Inherently elastic, globally distributed content delivery and request filtering.	No agency-managed scaling required; scales automatically with global demand.
Web and application tier	Auto Scaling Groups adjusting EC2 instance count, or equivalent container-based scaling.	CPU utilization, request count, or custom application metrics crossing defined thresholds.

Caching and async tier	ElastiCache cluster scaling and SQS/SNS queue depth-based scaling of downstream consumers.	Cache hit rate degradation or growing queue depth.
Database tier	Aurora read replica scaling and, where applicable, Aurora Serverless capacity adjustment.	Read replica lag, connection count, or CPU utilization thresholds.
Storage tier	Inherently elastic S3 storage with lifecycle and intelligent tiering policies.	No agency-managed scaling required; capacity is effectively unbounded.

► 3.2 Why Tier Independence Matters

- **Targeted scaling:** A demand spike primarily affecting read-heavy database queries can be addressed by scaling database read replicas without necessarily requiring proportional scaling of the web and application tier.
- **Independent optimization:** Isolating scaling decisions by tier allows each tier's cost and performance tradeoffs, examined in Section 4, to be optimized independently rather than as a single, undifferentiated scaling decision.

IV. CAPACITY STRATEGY COMPARISON AND COST-PERFORMANCE TRADEOFFS

Agencies choose among several distinct capacity provisioning strategies, each carrying a different cost and performance profile.

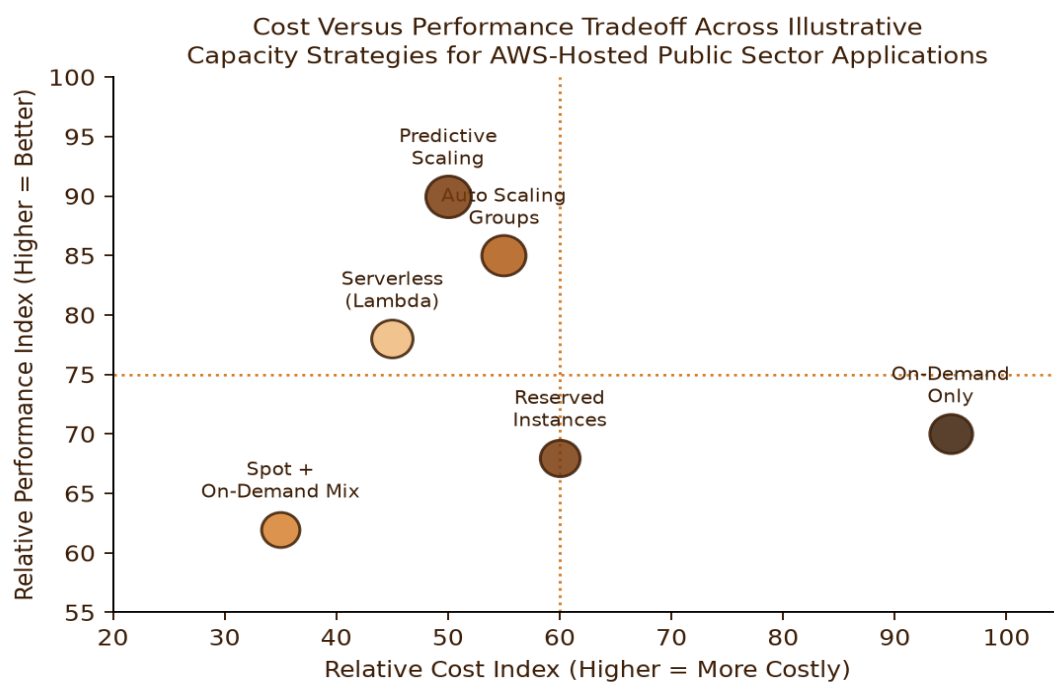


Figure 3. Cost versus performance tradeoff across illustrative capacity strategies for AWS-hosted public sector applications.

► 4.1 Strategy Descriptions

Strategy	Description	Best Suited Demand Pattern
On-demand only	Instances provisioned and billed purely on-demand with no scaling automation.	Not well suited to any of the demand patterns in Table 2; included as a baseline comparison.
Reserved instances	Capacity reserved in advance at a discounted rate for a committed term.	Predictable cyclical demand with a well understood baseline.
Auto scaling groups	Automated instance count adjustment based on defined utilization metrics.	Announced but variable-magnitude demand and unannounced demand spikes.

Spot plus on-demand mix	A blend of lower-cost, interruptible spot capacity with on-demand capacity as a fallback.	Workloads are tolerant of interruption, often batch or asynchronous processing rather than user-facing request handling.
Serverless (Lambda)	Function-based compute billed per invocation, with no server management or idle capacity cost.	Highly variable, spiky demand with unpredictable idle periods.
Predictive scaling	Auto scaling informed by forecasted demand rather than reactive metric thresholds alone.	Announced demand events with some lead time for forecast-informed pre-scaling.

► 4.2 Interpreting the Cost-Performance Tradeoff

As illustrated in Figure 3, on-demand only provisioning sits in the high-cost, moderate-performance quadrant, reflecting the absence of any discount mechanism or elasticity benefit. Predictive scaling and auto scaling groups occupy the more favorable lower-cost, higher-performance region, consistent with their ability to match provisioned capacity to actual demand more precisely than static approaches.

- **Blended strategy guidance:** No single strategy in Table 5 is universally optimal; most mature capacity plans combine several strategies across different workload types and tiers, for example serverless for spiky asynchronous processing alongside auto scaling groups for the primary web tier.

V. LATENCY AND UTILIZATION PATTERNS ACROSS TIME

Latency and utilization vary predictably across hour of day and day of week for most public sector applications, providing a valuable signal for both auto scaling threshold tuning and scheduled scaling policy design.

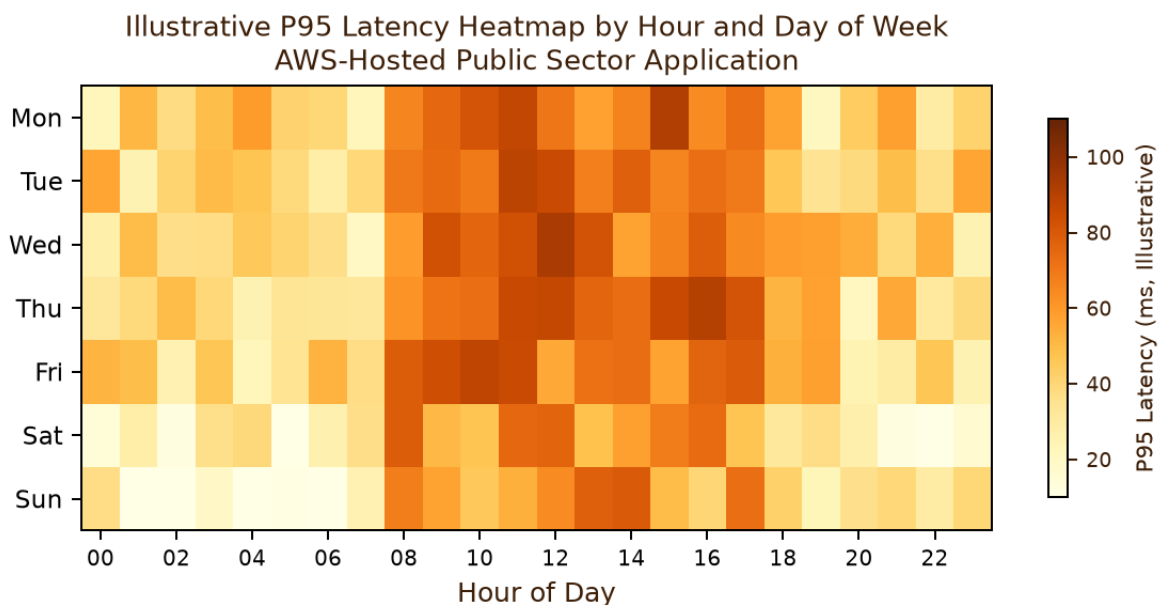


Figure 4. Illustrative P95 latency heatmap by hour and day of week for an AWS-hosted public sector application.

► 5.1 Pattern Observations

- **Business hours elevation:** Latency is illustrated during business hours, roughly 08:00 to 18:00, on weekdays, consistent with typical public sector application usage concentrated during business hours when caseworkers and applicants are most active.
- **Weekend reduction:** Weekend latency is markedly lower than weekday latency, reflecting reduced usage during Saturday and Sunday for most public sector applications outside of specific always-on services.
- **Threshold design implication:** The heatmap pattern in Figure 4 illustrates why uniform, time-independent auto scaling thresholds may be suboptimal; thresholds calibrated for weekday peak hours may be overly conservative, and thus costly, if applied uniformly across weekend low-usage periods.

► 5.2 Using Time-Based Patterns in Capacity Planning

Time-Based Pattern	Capacity Planning Response
Predictable weekday business-hours peak	Scheduled scaling actions pre-provisioning capacity ahead of the typical peak window, reducing reliance on reactive scaling alone.
Predictable weekend reduction	Scheduled scale-down actions reducing provisioned capacity during low-usage weekend windows to reduce cost.
Anomalous deviation from the typical pattern	Deviations from the established heatmap pattern, such as elevated weekend traffic, can serve as an early indicator of an emerging, atypical demand event requiring capacity review.

VI. DATABASE TIER CAPACITY CONSIDERATIONS

The database tier frequently becomes the binding capacity constraint during demand spikes, given that database scaling is generally slower and more operationally complex than compute tier scaling.

► 6.1 Database Scaling Approaches

Approach	Description	Scaling Speed	Common Limitation
Vertical scaling (instance resize)	Increasing the compute and memory capacity of a single database instance.	Slow; typically requires a brief outage or failover.	Upper bound limited by largest available instance size.
Read replica scaling	Adding read replicas to distribute read query load away from the primary writer instance.	Moderate replica provisioning takes minutes.	Does not address write-heavy workload bottlenecks.
Aurora Serverless capacity scaling	Automatic capacity adjustment within a defined range based on load.	Fast, typically within seconds for capacity within the configured range.	Range must be configured with sufficient headroom in advance.
Read/write workload separation	Architecting the application to route read and write queries to different database endpoints.	Not a scaling mechanism itself, but enables more effective use of read replica scaling.	Requires application-level query routing logic.

► 6.2 Database Tier Capacity Planning Priorities

- **Priority 1:** Confirm that read replica scaling is architecturally usable, meaning the application already separates read and write traffic, before relying on read replica scaling as a primary spike response mechanism.
- **Priority 2:** Configure Aurora Serverless capacity ranges, where used, with sufficient headroom informed by the demand multiplier estimates in Table 2 rather than narrow ranges sized only for typical baseline demand.
- **Priority 3:** Treat vertical scaling as a planned, scheduled capacity change ahead of a known demand event rather than a reactive response during an active spike, given its slower scaling speed.

VII. CACHING AND ASYNCHRONOUS PROCESSING STRATEGIES

Caching and asynchronous processing reduce the load reaching the database and compute tiers respectively, extending effective capacity without proportional infrastructure cost increase.

► 7.1 Caching Strategy Comparison

Caching Strategy	Description	Best Suited Data Type
Application-level caching (ElastiCache)	Frequently accessed data cached in-memory, reducing repeated database queries.	Reference data and frequently read, infrequently updated case or program data.
Content delivery network caching (CloudFront)	Static and semi-static content cached at edge locations close to end users.	Static assets, public program information pages, and cacheable API responses.
Database query result caching	Query results cached at the database or application layer for a defined time-to-live period.	Computationally expensive queries with acceptable staleness tolerance.

► 7.2 Asynchronous Processing Strategy Comparison

Asynchronous Strategy	Description	Best Suited Workload
Message queue-based decoupling (SQS)	Time-insensitive processing tasks placed on a queue and processed independently of the originating request.	Document processing, notification generation, and batch eligibility recalculation.
Event-driven fan-out (SNS)	A single event triggers multiple independent downstream processing paths.	Workflows requiring multiple, independent follow-up actions from a single triggering event.
Scheduled batch processing	Non-urgent processing deferred to scheduled, off-peak execution windows.	Reporting, data quality validation, and other work tolerant of processing delay.

◆ PERFORMANCE NOTE

Caching and asynchronous processing should be viewed as capacity multipliers rather than substitutes for the multi-tier scaling architecture in Section 3; they extend the effective capacity of a given infrastructure footprint but do not eliminate the need for elastic scaling during sufficiently large demand spikes.

VIII. COST COMPOSITION AND OPTIMIZATION LEVERS

Understanding how cost is distributed across AWS service categories over the application lifecycle informs where optimization effort yields the greatest return.

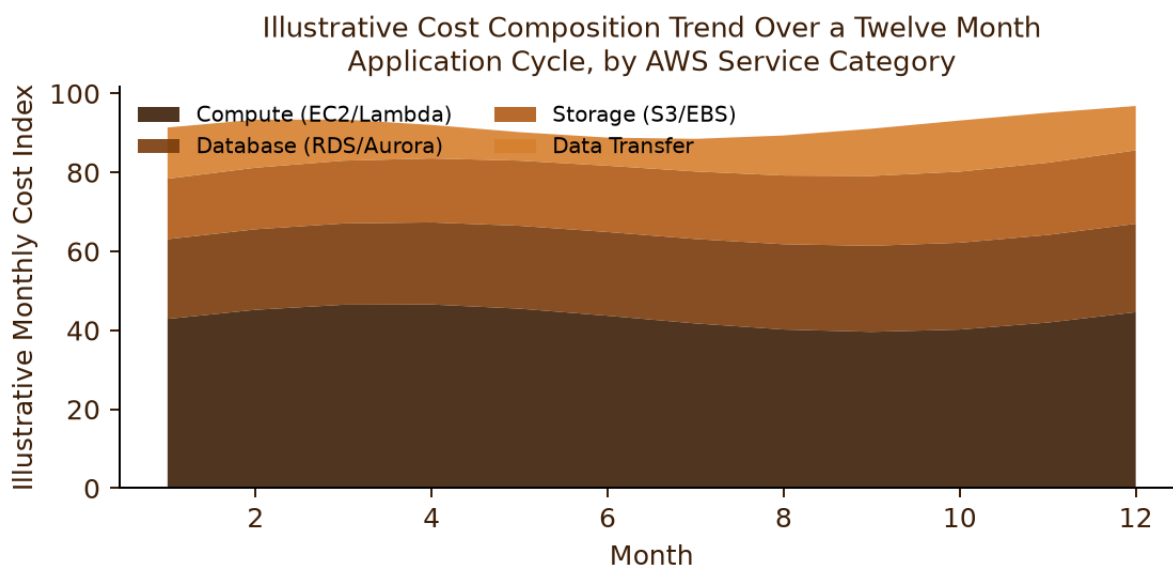


Figure 5. Illustrative cost composition trend over a twelve month application cycle, by AWS service category.

► 8.1 Cost Category Optimization Levers

Cost Category	Primary Optimization Lever	Related Section
Compute (EC2/Lambda)	Right-sizing instance types and Lambda memory allocation; blending capacity strategies per Section 4.	Sections 3 and 4
Database (RDS/Aurora)	Right-sizing instance classes; leveraging Aurora Serverless capacity scaling where appropriate.	Section 6
Storage (S3/EBS)	Lifecycle policies and intelligent tiering to move infrequently accessed data to lower-cost storage classes.	Not separately detailed; addressed via standard AWS storage lifecycle guidance.
Data transfer	Minimizing cross-region and cross-availability-zone transfer through architecture design; leveraging CloudFront caching.	Section 7.1

► 8.2 Optimization Practice Scores

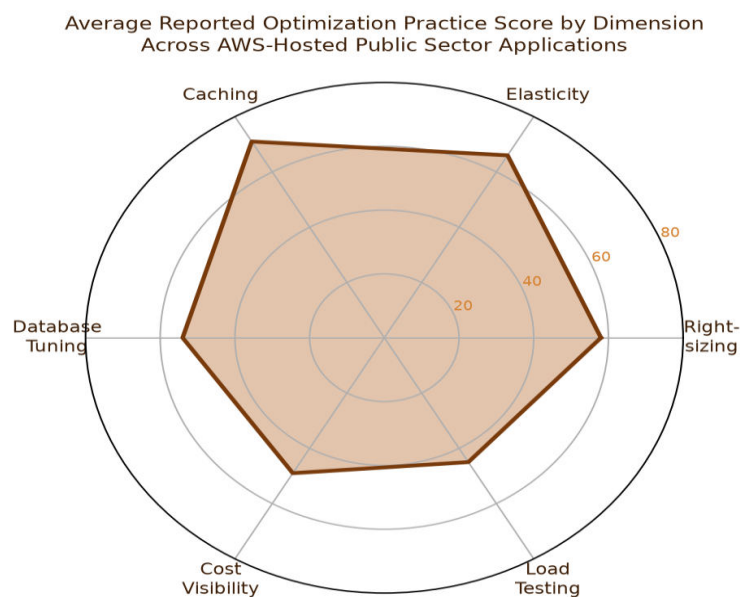


Figure 6. Average reported optimization practice score by dimension across AWS-hosted public sector applications, illustrative scale of 0 to 100.

- **Highest scoring dimension:** Caching, at 71 of 100, scored highest among the six optimization dimensions illustrated, suggesting relatively mature caching adoption compared to other dimensions.
- **Lowest scoring dimension:** Load testing, at 45 of 100, scored lowest, suggesting that many agencies size capacity without systematically validating that sizing against realistic simulated demand, a gap explored further in Section 10.
- **Cost visibility gap:** Cost visibility, at 49 of 100, scored second lowest, indicating that many agencies lack granular, real-time visibility into which cost category is driving spend at a given time, limiting their ability to act on the cost composition patterns illustrated in Figure 5.

IX. THE FIVE STAGE CAPACITY PLANNING MATURITY MODEL

Agencies can be classified into one of five maturity stages describing their overall capacity planning and performance optimization readiness.

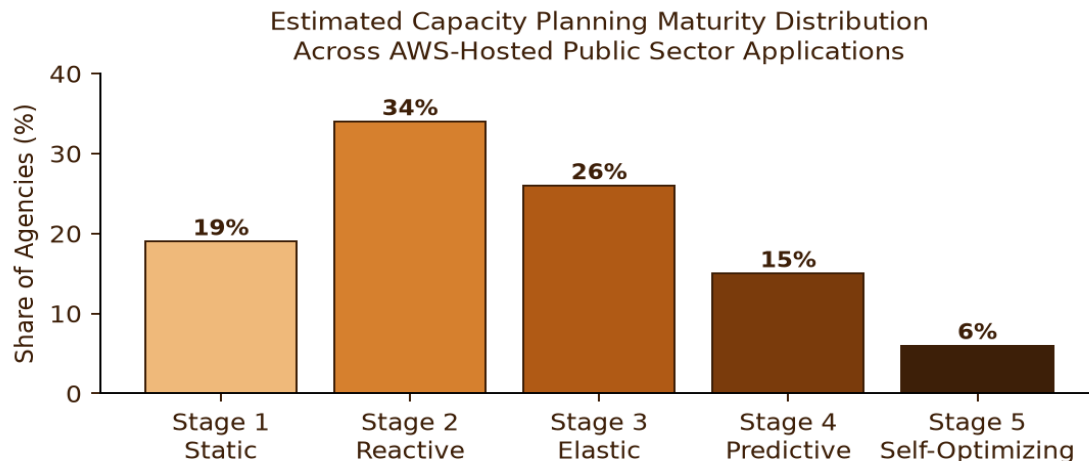


Figure 7. Estimated capacity planning maturity distribution across AWS-hosted public sector applications.

Stage	Name	Description
1	Static	Capacity is manually provisioned and rarely adjusted; no auto scaling is configured.
2	Reactive	Basic auto scaling exists but is configured reactively, often after an incident, without systematic tuning.
3	Elastic	Auto scaling is systematically configured across most tiers per the reference architecture in Section 3.
4	Predictive	Scaling incorporates demand forecasting per Section 10, informed by the time-based patterns in Section 5.
5	Self-Optimizing	Capacity and cost optimization operate continuously, informed by the KPIs in Section 15, with minimal manual intervention.

X. LOAD TESTING AND DEMAND FORECASTING

Given the low load testing optimization score observed in Section 8.2, systematic load testing and demand forecasting represent a significant improvement opportunity for most agencies examined.

► 10.1 Load Testing Approaches

Load Testing Approach	Description	Best Suited Purpose
Baseline load testing	Testing under expected typical demand to confirm standard performance benchmarks.	Validating standard operating performance against service level objectives.
Spike testing	Testing under a sudden, large demand increase to validate auto scaling responsiveness.	Validating the demand exceedance scenarios illustrated in Figure 1.
Soak testing	Sustained testing over an extended duration to detect gradual degradation such as memory leaks.	Validating stability during extended high-demand periods.
Breakpoint testing	Progressively increasing load until failure to identify the actual capacity ceiling.	Establishing a validated, rather than assumed, capacity ceiling for planning purposes.

► 10.2 Demand Forecasting Approaches

- **Historical trend extrapolation:** Extrapolating recent historical demand trends forward, most effective for the predictable cyclical demand pattern category described in Table 3.
- **Event-informed forecasting:** Incorporating known upcoming policy or program events, such as an announced deadline, into capacity planning ahead of the event, addressing the announced but variable-magnitude demand pattern category.
- **Reactive playbook for unforecastable events:** Establishing a rapid-response capacity playbook for unannounced demand events, since these are inherently resistant to advance forecasting and require fast reactive scaling instead, per the auto scaling and predictive scaling strategies in Section 4.1.

XI. GOVERNANCE AND OPERATIONAL OWNERSHIP

Sustained capacity planning effectiveness requires clear governance ownership spanning architecture, operations, and cost management functions.

► 11.1 Governance Roles

Governance Role	Primary Responsibility	Related Section
Capacity engineering lead	Owns the multi-tier scaling architecture and auto scaling configuration.	Section 3
Cost optimization analyst	Monitors cost composition trends and drives right-sizing and cost visibility improvements.	Section 8
Performance testing lead	Owns the load testing program and demand forecasting inputs.	Section 10
Database capacity owner	Owns database tier scaling decisions and capacity range configuration.	Section 6

► 11.2 Common Governance Gaps

- Auto scaling thresholds configured at initial launch are frequently left unreviewed as actual usage patterns, such as those illustrated in Figure 4, diverge from initial assumptions over time.
- Cost optimization is often treated as a periodic, one-time review rather than an ongoing, continuously monitored practice, allowing cost composition drift, as illustrated in Figure 5, to go unaddressed.
- Load testing, consistent with its low optimization score in Section 8.2, is frequently conducted only once before initial launch rather than repeated as the application and its usage patterns evolve.

XII. RISK ASSESSMENT AND MITIGATION PLANNING

Capacity planning and performance optimization programs carry risks that agencies should evaluate and mitigate deliberately given the public facing consequences of capacity failure during high-stakes demand events.

Risk	Likelihood	Impact	Mitigation Approach
Database tier becomes the binding constraint during a spike	High	High	Validate read/write separation and configure adequate capacity headroom (Section 6.2).
Auto scaling thresholds miscalibrated for actual usage patterns	Moderate to High	High	Recalibrate thresholds using the time-based patterns in Section 5.2.
Untested capacity ceiling fails during an actual demand event	Moderate	High	Conduct breakpoint testing per Section 10.1 to establish a validated ceiling.
Cost composition drift goes undetected	High	Moderate	Establish continuous cost visibility monitoring rather than periodic review (Section 11.2).
Reliance on a single capacity strategy unsuited to the actual demand pattern	Moderate	Moderate	Blend capacity strategies according to workload type per Section 4.2.

► **12.1 Risk Prioritization Guidance**

- Database tier and untested capacity ceiling risks should be treated as highest priority given their direct connection to public-facing service failure during exactly the high-visibility demand events capacity planning is meant to address.
- Auto scaling threshold miscalibration risk compounds over time and should be addressed through the recurring governance cadence described in Section 13.
- Cost composition drift, while lower in direct service impact, carries significant budgetary consequence over a multi-year application lifecycle if left unmonitored.

XIII. IMPLEMENTATION ROADMAP

Translating this framework into an executable program requires sequencing architecture assessment, testing, and governance formation.

Phase	Approximate Timeframe	Key Milestones
Architecture Assessment	Months 1 to 2	Assess current architecture against the multi-tier reference design in Section 3; identify tier-level scaling gaps.
Load Testing and Baseline Establishment	Months 2 to 4	Conduct baseline, spike, and breakpoint testing per Section 10.1 to establish a validated capacity ceiling.
Auto Scaling and Caching Implementation	Months 3 to 7	Implement or refine auto scaling configuration and caching strategies per Sections 3 and 7.
Cost Optimization and Governance Formation	Months 6 to 9	Establish the governance roles in Section 11.1 and continuous cost visibility monitoring.
Predictive Scaling and Continuous Optimization	Months 9 onward	Incorporate demand forecasting per Section 10.2 and operate the KPIs described in Section 15.

► **13.1 Governance Cadence Reference**

- **Weekly during active monitoring:** Review cost composition trends and auto scaling activity.
- **Monthly:** Review latency and utilization patterns against the time-based benchmarks in Section 5.
- **Quarterly to annually:** Conduct load testing refresh and a full capacity planning maturity reassessment.

XIV. CASE PATTERNS AND INDUSTRY BENCHMARKS

This article synthesizes recurring patterns observed across published AWS architectural guidance and public sector cloud performance research through mid 2023, rather than reporting on a single named agency implementation.

Pattern Observed	Reported Association
Multi-tier independent scaling architecture	Reduced binding constraint incidents at any single tier during demand spikes (Section 3.2).
Blended capacity strategy matched to workload type	Improved cost-performance positioning relative to single-strategy approaches (Figure 3).
Scheduled scaling informed by time-based usage patterns	Reduced cost during predictable low-usage periods without sacrificing peak-hour performance (Section 5.2).
Systematic breakpoint and spike load testing	Fewer unvalidated capacity assumptions carried into production (Section 10.1).
Continuous cost visibility monitoring	Earlier detection of cost composition drift compared to periodic review alone (Section 11.2).

► **14.1 Cross-Program Observations**

- **Observation 1:** Programs subject to the unannounced, externally triggered demand pattern category in Table 3, such as unemployment insurance during economic downturns, reported the greatest benefit from fully elastic, rapid-response scaling architecture.

- **Observation 2:** Programs with well-established, predictable annual cycles, such as recurring renewal periods, benefited disproportionately from scheduled scaling and demand forecasting relative to programs with more erratic demand patterns.
- **Observation 3:** Agencies with a dedicated cost optimization analyst role, per Section 11.1, reported more consistent cost composition monitoring than agencies where cost review was an informal, collateral responsibility.

XV. KEY PERFORMANCE INDICATORS FOR ONGOING GOVERNANCE

Sustaining capacity planning and performance optimization value requires an ongoing measurement framework spanning latency, cost, and scaling responsiveness.

KPI	Definition	Illustrative Target
P95 and P99 latency	95th and 99th percentile response latency, tracked against the time-based patterns in Figure 4.	Stable across both typical and peak demand periods.
Auto scaling response time	Time from a scaling trigger event to fully provisioned additional capacity.	Short enough to prevent sustained capacity exceedance during a spike.
Capacity exceedance frequency	Number of instances where observed demand exceeded provisioned capacity, per the pattern in Figure 1.	Declining over successive reporting periods.
Cost per transaction	Total infrastructure cost divided by transaction volume, normalizing cost trends against usage.	Stable or declining even as absolute usage grows.
Load test coverage recency	Time since the most recent spike and breakpoint load test.	Refreshed on the quarterly cadence described in Section 13.1.
Capacity planning maturity stage progression	Composite score tracked against the five stage model in Section 9.	Steady progression toward higher maturity stages.

◆ GOVERNANCE REMINDER

KPIs should be reviewed on the weekly, monthly, and quarterly cadence described in Section 13.1, with particular attention to whether cost per transaction is declining even as absolute usage grows, the clearest signal that optimization efforts are outpacing raw demand growth.

XVI. CONCLUSION AND RECOMMENDATIONS

Capacity planning and performance optimization for AWS-hosted public sector enterprise applications requires accounting for demand dynamics distinct from typical commercial application patterns. Agencies that adopt multi-tier elastic architecture, blend capacity strategies to match workload type, and govern capacity decisions continuously report substantially better cost efficiency and latency stability than agencies relying on static, manually provisioned capacity.

► 16.1 Summary of Key Findings

- Public sector demand spikes, driven by policy, economic, and disaster-related triggers, can exceed baseline traffic by an order of magnitude, exceeding what static provisioning can accommodate.
- No single capacity strategy is universally optimal; blended strategies matched to workload type consistently outperform single-strategy approaches on the cost-performance tradeoff.
- Load testing and cost visibility are the lowest-scoring optimization dimensions observed, representing the most significant improvement opportunity for most agencies.
- The database tier frequently becomes the binding capacity constraint during demand spikes, given its comparatively slower and more operationally complex scaling characteristics.

► 16.2 Recommendations

- **Recommendation 1:** Adopt the multi-tier scaling architecture in Section 3, ensuring each tier scales independently according to its own utilization characteristics.
- **Recommendation 2:** Blend capacity strategies according to workload type rather than relying on a single strategy across all compute needs, per the comparison in Section 4.1.
- **Recommendation 3:** Establish a systematic, recurring load testing program, including breakpoint testing, to replace unvalidated capacity assumptions with tested capacity ceilings.
- **Recommendation 4:** Invest in continuous cost visibility monitoring and a dedicated cost optimization governance role to prevent the cost composition drift risk described in Section 12.

REFERENCES

- [1] Amazon Web Services. (2022). "AWS Well-Architected Framework: Performance Efficiency Pillar." AWS Whitepaper.
- [2] Amazon Web Services. (2022). "AWS Well-Architected Framework: Cost Optimization Pillar." AWS Whitepaper.
- [3] Amazon Web Services. (2021). "AWS Well-Architected Framework: Reliability Pillar." AWS Whitepaper.
- [4] Amazon Web Services. (2022). "Best Practices for Amazon RDS and Aurora Performance." AWS Technical Documentation.
- [5] Amazon Web Services. (2021). "Amazon ElastiCache Best Practices." AWS Technical Documentation.
- [6] Amazon Web Services. (2022). "Auto Scaling Best Practices Guide." AWS Technical Documentation.
- [7] FinOps Foundation. (2022). "FinOps Framework: Cloud Financial Management Principles and Practices." FinOps Foundation Publication.
- [8] Beyer, B., Jones, C., Petoff, J., & Murphy, N.R. (2016). "Site Reliability Engineering: How Google Runs Production Systems." O'Reilly Media.
- [9] Beyer, B., Murphy, N.R., Rensin, D.K., Kawahara, K., & Thorne, S. (2018). "The Site Reliability Workbook: Practical Ways to Implement SRE." O'Reilly Media.
- [10] U.S. Government Accountability Office. (2021). "Cloud Computing: Agencies Have Increased Usage and Realized Benefits, but Face Challenges." GAO Report.
- [11] U.S. Government Accountability Office. (2020). "Unemployment Insurance: States' Information Technology Systems Faced Challenges Responding to the Pandemic." GAO-21-262.
- [12] National Association of State Chief Information Officers (NASCIO). (2022). "State CIO Top Ten Policy and Technology Priorities." NASCIO Research Brief.
- [13] Center for Digital Government. (2022). "Cloud Cost Optimization and Performance Management in State Government." Center for Digital Government Research Report.
- [14] Kleppmann, M. (2017). "Designing Data-Intensive Applications." O'Reilly Media.
- [15] Newman, S. (2021). "Building Microservices," 2nd Edition. O'Reilly Media.
- [16] Government Technology. (2022). "States Rethink Cloud Capacity Planning After Pandemic-Era Demand Spikes." e.Republic Government Technology Publication.
- [17] Forsgren, N., Humble, J., & Kim, G. (2018). "Accelerate: The Science of Lean Software and DevOps." IT Revolution Press.
- [18] Gartner, Inc. (2022). "Market Guide for Cloud Financial Management Tools." Gartner Research.



Impact Factor: 8.423



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  ijirset@gmail.com



www.ijirset.com

Scan to save the contact details