System Design Component Implementation

Report

1. Test Execution Summary

All automated tests completed successfully. The test suite ran 12 tests across 2 test files (simulation.test.ts and api.test.ts), with 0 failures.

Reported summary:

- Test Suites: 2 passed / 2 total
- Tests: 12 passed / 12 total
- Snapshots: 0
- Execution time: ≈ 1.1 seconds

This confirms both the API layer and the system simulation logic behave as expected under the current scenarios.

2. Resource Usage Analysis

The simulation was executed for four runs to observe resource consumption patterns for the main components:

- IdentityProviderClient
- SmartMachineClient
- MachineStateTable
- DataCache

Across all four runs, resource usage was very stable:

- IdentityProviderClient
  - Units: 1,280,512 per run
  - Share: ~93.7–93.9% of total work
  - Dominates overall cost and is the main bottleneck under this workload.
- MachineStateTable
  - Units: 55,304 per run
  - Share: ~4.05% of total work
  - Second-largest contributor, but an order of magnitude smaller than IdentityProviderClient.
- DataCache
  - Units: 22,608 per run
  - Share: ~1.65–1.66% of total work
- SmartMachineClient
  - Units: 6,912 per run
  - Share: ~0.51% of total work

The near-identical values across the four simulations indicate deterministic, repeatable resource allocation under the given test conditions. IdentityProviderClient and MachineStateTable together account for roughly 98% of total simulated work.

3. Cache Performance Metrics

The simulation logs also report cache hit and miss statistics for each run:

- Run 1: 3,914 hits, 2,159 misses – hit rate ≈ 64.45%
- Run 2: 3,913 hits, 2,057 misses – hit rate ≈ 65.54%
- Run 3: 3,816 hits, 2,167 misses – hit rate ≈ 63.78%
- Run 4: 3,876 hits, 2,120 misses – hit rate ≈ 64.64%

Cache hit rate consistently falls in a narrow band of about 63–66%. This suggests that, for the modeled access pattern, the cache successfully serves most lookups while still leaving a significant fraction to fall through to the backing store.

The variation between the best and worst hit rate is under 2 percentage points, indicating stable caching behavior across repeated executions with the same workload.

4. Database Access Patterns

A final table summarizes cache hits relative to total database accesses:

- DB accesses per run: 6,624 (constant across all runs)
- Hit/Access ratios:
  - Run 1: 0.5909
  - Run 2: 0.5907
  - Run 3: 0.5761
  - Run 4: 0.5851

These ratios mean that for each database access in the simulation, there are roughly 0.58–0.59 cache hits. The DB access count remaining constant confirms that the simulated workload is identical each run, while the tight clustering of the hit/access ratio reflects predictable interaction between the cache and the backing store.

5. Observations

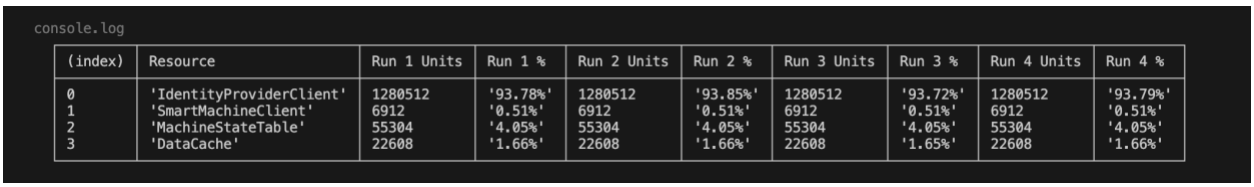
- The system exhibits highly consistent behavior across four simulation runs: resource units, percentages, and DB access counts are effectively identical.
- IdentityProviderClient is the dominant consumer, suggesting that any future optimization effort should first target reducing calls to (or cost of) this component.
- The cache delivers a moderate performance benefit (~64% hit rate) while still leaving substantial load on the database, which could be improved by tuning cache size, eviction strategy, or access patterns.

6. Conclusion

All tests passed successfully, and the simulation produced stable, repeatable metrics for resource usage, caching, and database activity. Resource consumption is heavily skewed toward the IdentityProviderClient, while the cache consistently serves about two-thirds of lookups. Overall, these results provide a reliable baseline characterization of

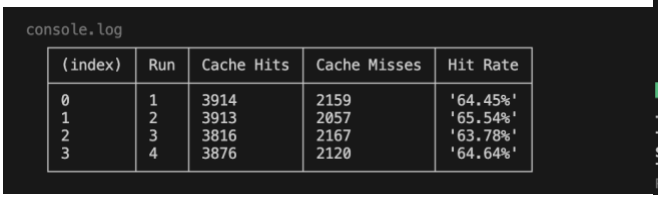
the system's behavior under the current simulated workload and validate the correctness of both the API and simulation components.

Simulation test screenshots:



console.log

(index)	Resource	Run 1 Units	Run 1 %	Run 2 Units	Run 2 %	Run 3 Units	Run 3 %	Run 4 Units	Run 4 %
0	'IdentityProviderClient'	1280512	'93.78%'	1280512	'93.85%'	1280512	'93.72%'	1280512	'93.79%'
1	'SmartMachineClient'	6912	'0.51%'	6912	'0.51%'	6912	'0.51%'	6912	'0.51%'
2	'MachineStateTable'	55304	'4.05%'	55304	'4.05%'	55304	'4.05%'	55304	'4.05%'
3	'DataCache'	22608	'1.66%'	22608	'1.66%'	22608	'1.65%'	22608	'1.66%'



console.log

(index)	Run	Cache Hits	Cache Misses	Hit Rate
0	1	3914	2159	'64.45%'
1	2	3913	2057	'65.54%'
2	3	3816	2167	'63.78%'
3	4	3876	2120	'64.64%'

console.log

(index)	Run	Cache Hits	DB Accesses	Hit/Access Ratio
0	1	3914	6624	'0.5909'
1	2	3913	6624	'0.5907'
2	3	3816	6624	'0.5761'
3	4	3876	6624	'0.5851'

at Object.<anonymous> (test/simulation.test.ts:162:13)

PASS test/api.test.ts

Test Suites: 2 passed, 2 total
Tests: 12 passed, 12 total
Snapshots: 0 total
Time: 1.098 s
Ran all test suites.