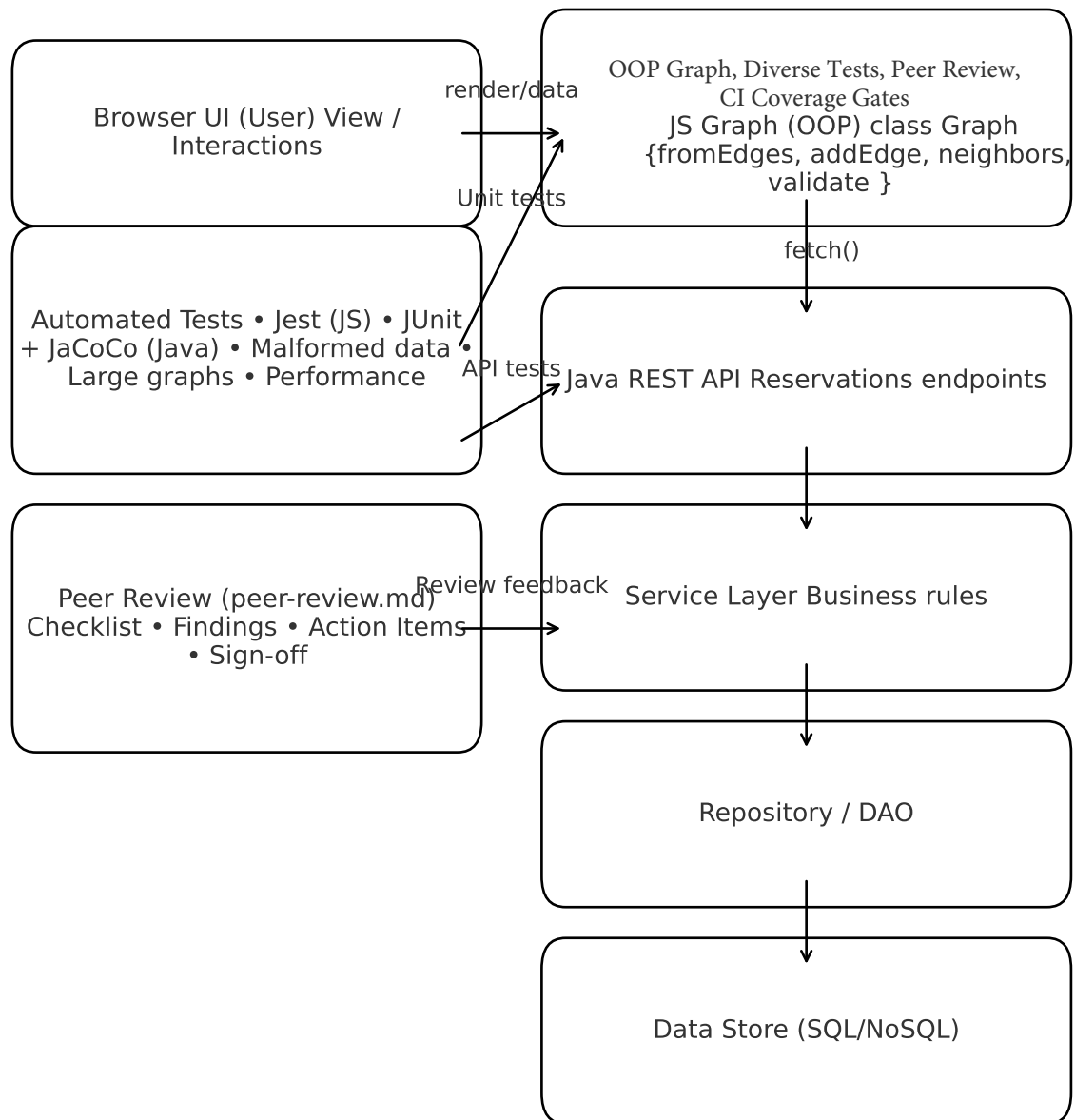


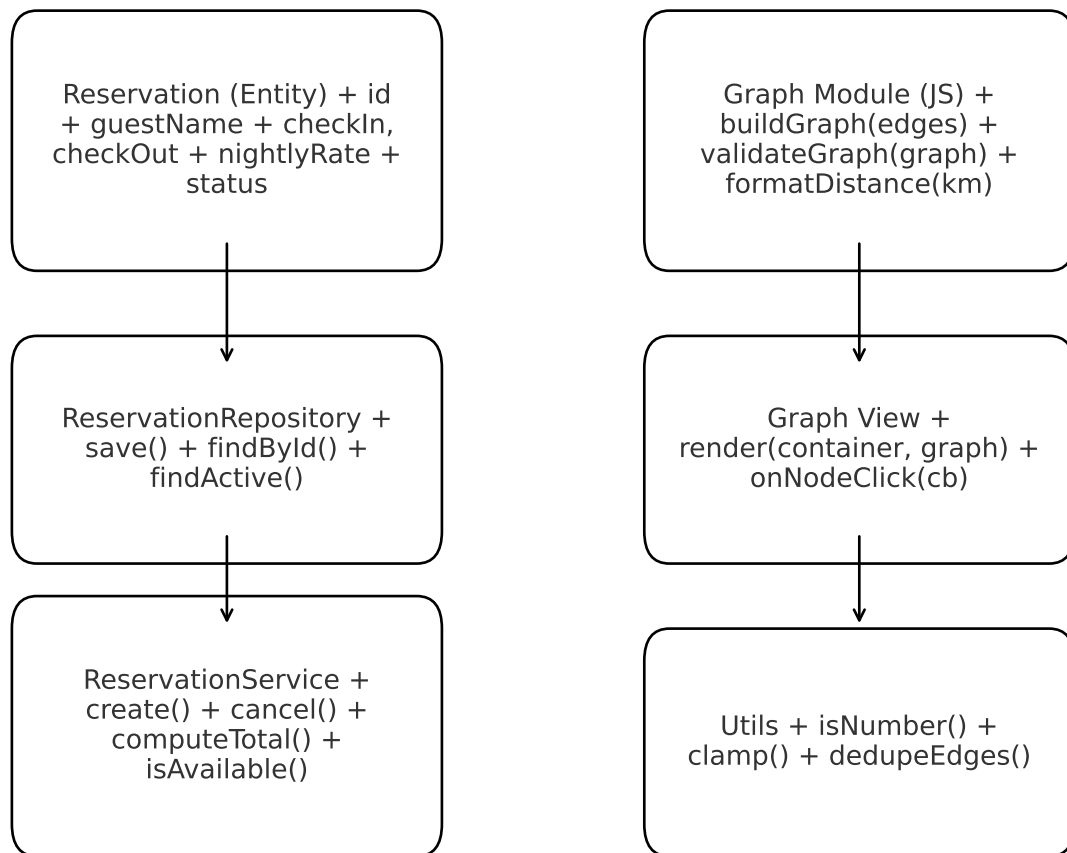
Challenge 7 System Overview



Notes:

- JS graph refactored to an OOP class for reusability/testability.
- Diverse tests improve robustness and performance confidence.
- Peer review log documents collaborative validation.
- Clear separation of concerns across layers.

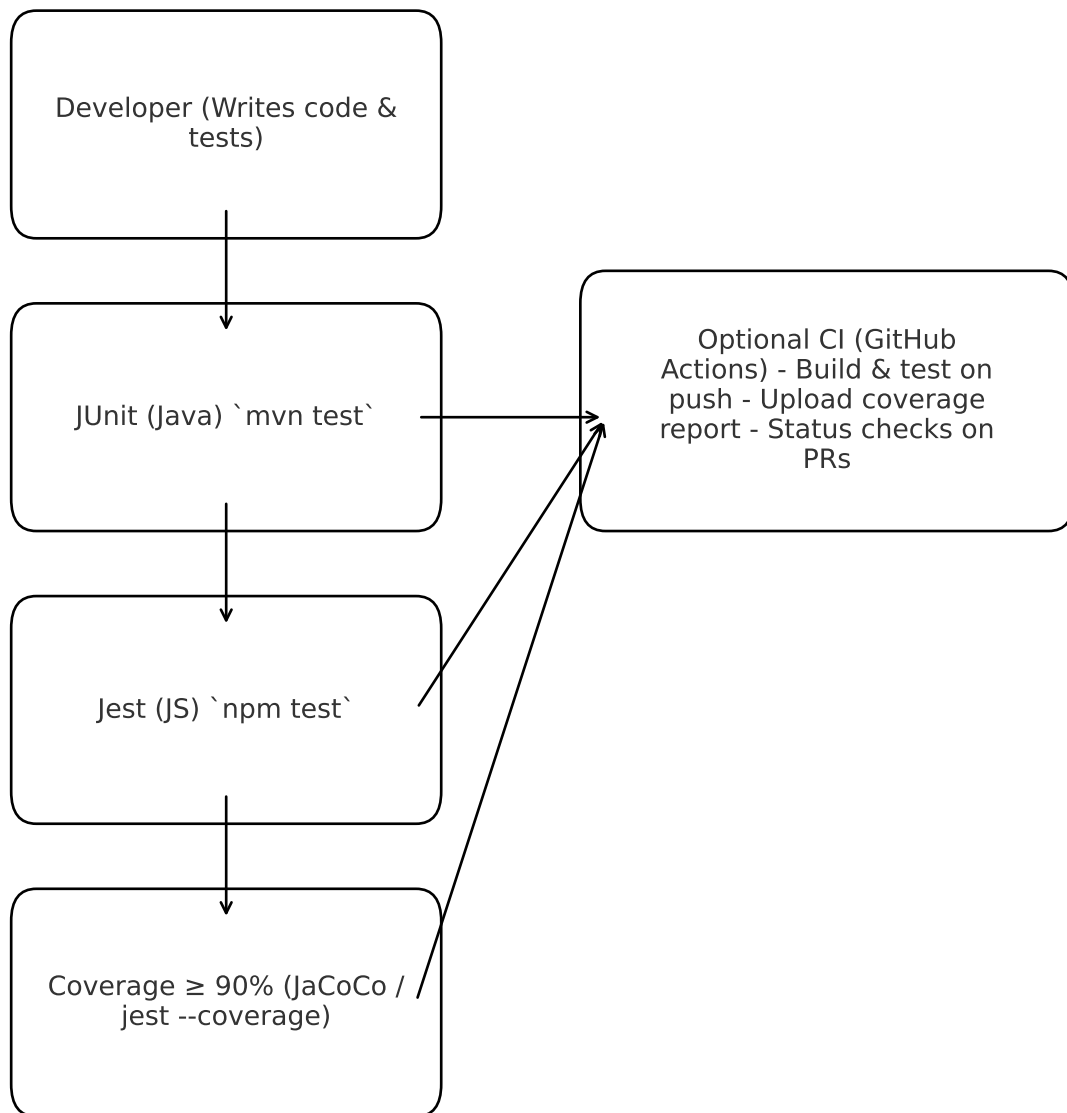
Module & Class Overview (Simplified)



Key Decisions:

- Services enforce rules; Repository isolates persistence.
- JS module is framework-agnostic; pure functions are testable.

Test & CI Flow



JS Graph Class Design (v3)

Graph — adj: Map<string, Map<string, number>> — static fromEdges(edges, {validate}) — addEdge(u, v, km) [undirected; min km] — neighbors(u) -> {city, km}[] — validate() [no self-loops; km finite, ≥ 0] — size (getter), edgeCount (getter)

Policies • Ignore self-loops • Keep smallest parallel edge • Throw on invalid inputs

Usage import Graph from './src/graph_oop.js' const g = Graph.fromEdges(edges) view.render('#container', g)

Test Coverage Targets • Malformed inputs • Large rings (1,000 nodes) • Dense clusters (~19,900 edges) • Timing thresholds are soft (environment dependent)

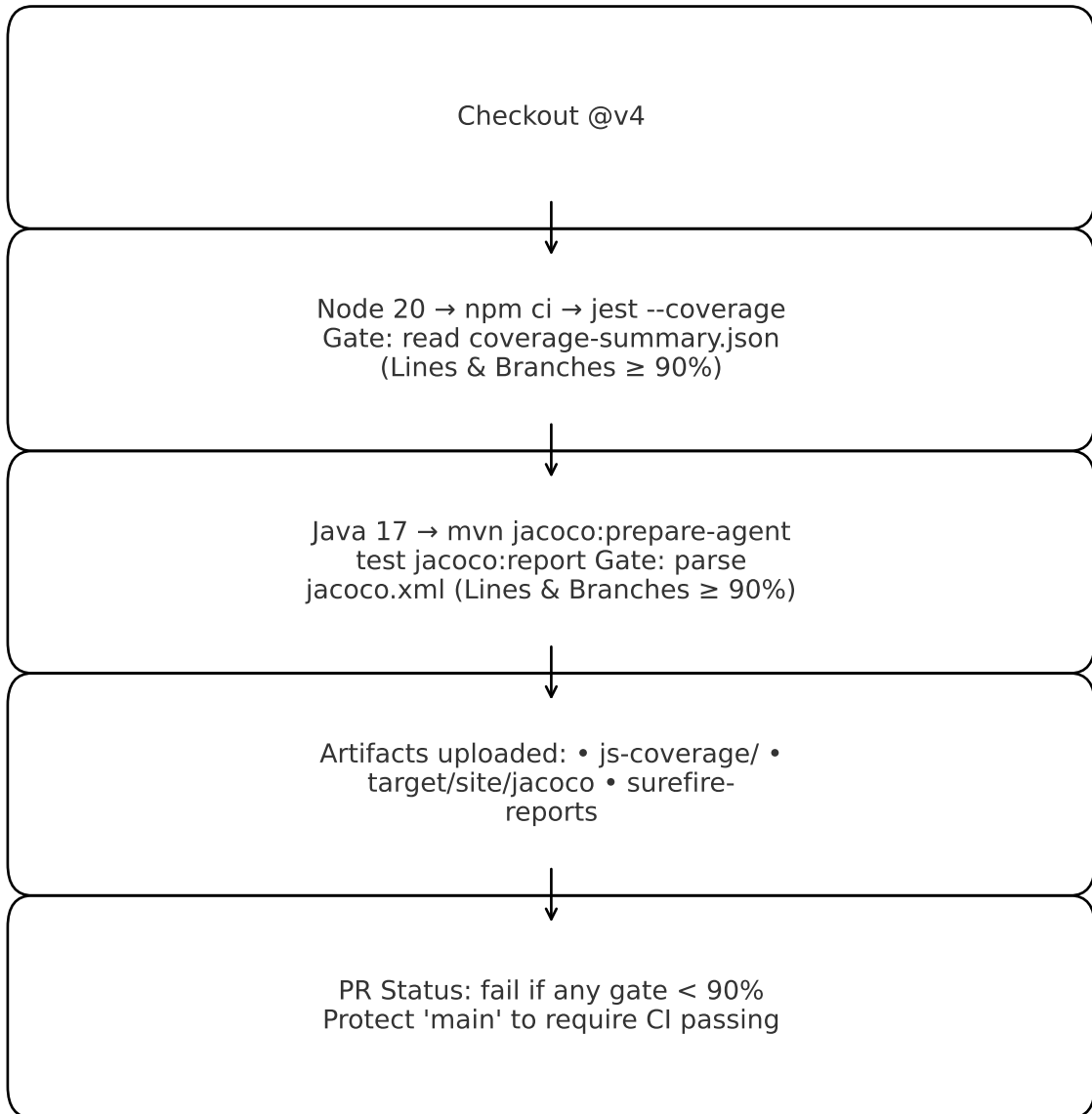
Automated Tests Matrix

Jest (JS) Files: graph.test.js,
graph.malformed.test.js,
graph.large.test.js,
graph.performance.test.js Asserts:
structure, invariants, edgeCount,
soft timing Coverage Gates: Lines \geq
90%, Branches \geq 90%

JUnit + JaCoCo (Java) Service &
Repository tests: create/cancel,
availability, totals, edge cases
JaCoCo report parsed from
jacoco.xml; coverage gates Lines \geq
90%, Branches \geq 90%

Execution Flow Dev \rightarrow npm test (Jest)
& mvn test (JUnit) If below
thresholds \rightarrow fix tests/logic \rightarrow re-
run Artifacts: js-coverage/ & jacoco
site

CI Workflow Coverage Gates (v3)



Repository Structure

```
/
├── README.md
├── README_addendum_feedback.md
├── peer-review.md
├── docs/
│   ├── Challenge7_Diagrams.pdf
│   ├── Evidence/
│   │   ├── js-coverage-console.png
│   │   ├── js-coverage-summary.png
│   │   ├── junit-console.png
│   │   └── jacoco-index.png
│   ├── Images/
│   │   └── (opcional: capturas o diagramas extra para el README)
│   └── .github/
│       ├── workflows/
│       │   └── ci.yml
│       └── java-reservations/
│           ├── pom.xml
│           └── src/
│               ├── main/
│               │   ├── java/
│               │   │   ├── com/
│               │   │   │   ├── bookingmx/
│               │   │   │   │   ├── reservations/
│               │   │   │   │   │   ├── model/
│               │   │   │   │   │   │   ├── Reservation.java
│               │   │   │   │   │   │   ├── repository/
│               │   │   │   │   │   │   │   ├── ReservationRepository.java
│               │   │   │   │   │   │   └── service/
│               │   │   │   │   │   │       └── ReservationService.java
│               │   └── test/
│               │   │   ├── java/
│               │   │   │   ├── com/
│               │   │   │   │   ├── bookingmx/
│               │   │   │   │   │   ├── reservations/
│               │   │   │   │   │   │   ├── ReservationServiceTest.java
│               │   │   │   │   │   │   └── (más pruebas JUnit)
│               │   └── target/
│               │   │   ├── site/
│               │   │   │   ├── jacoco/
│               │   │   │   │   ├── index.html
│               │   │   │   └── jacoco.xml
│               └── js-graph/
│                   ├── package.json
│                   ├── jest.config.js
│                   └── src/
│                       ├── graph_oop.js
│                       ├── graph.js
│                       └── view.js
│               └── __tests__/
│                   ├── graph.malformed.test.js
│                   ├── graph.large.test.js
│                   ├── graph.performance.test.js
│                   └── graph.test.js
```