SAMPLE REPORT · PUBLIC CODE

MODERNIZATION ASSESSMENT

spring-petclinic-rest v2.6.2

Findings, upgrade path to Java 21 and Spring Boot 4.1, and a fixed-price quote.

APPLICATION	SOURCE	CURRENT STACK
spring-petclinic-rest, release v2.6.2 (commit 517a399, 28 Dec 2021)	github.com/spring-petclinic/spring-petclinic-rest · Apache 2.0	Java 8 · Spring Boot 2.6.2
PREPARED		
30 October 2026 · Pashyaa Technologies, Pune		

This is a sample. We produced it on public open-source code to show what a Pashyaa Modernization Assessment delivers. It is not client work, and spring-petclinic-rest is a reference application maintained by the Spring community, not a production system. Every number below comes from running the build, tests and scans described in section 2. A paying client receives the same report on their own application, plus a walkthrough call.

1 Executive summary

OVERALL RISK: HIGH

The application builds cleanly and has a strong test suite, but it runs on a framework stack that has been out of open-source support for years and cannot move to Java 21 without a framework upgrade.

171 tests pass, 0 failures	**88.4%** line coverage (71.1% branch)
144 known advisories in shipped libraries, 19 critical	**$8,100** fixed price to Java 21, after credit

- **The build is healthy.** `mvn verify` passes on the first try: 171 tests, 0 failures, in about one minute. Line coverage is 88.4% and branch coverage is 71.1%. All tests already use JUnit 5. This is a good starting point and lowers the cost of the upgrade.
- **The dependencies carry 144 known security advisories** (19 critical, 57 high) in libraries that ship with the application, matched against the GitHub Advisory Database as of 30 Oct 2026. For 35 of them the advisory lists no fixed version on the current Spring 5.3 / Boot 2.x line, so only an upgrade removes them. 60 of the 144 were published in 2026 alone, so the exposure grows every month.
- **We checked the 19 critical advisories against the code.** 13 of them need a feature this application doesn't use (WAR deployment, WebFlux, Tomcat DIGEST login, Cloud Foundry) or are low risk here. The other 6 need deeper review, and the upgrade removes all 19. Several high-severity denial-of-service issues in the JSON parser (Jackson) do apply, because the API accepts JSON from callers.
- **Java 21 is blocked by the framework, not the code.** When we compiled the unchanged application for Java 21, 14 of 15 test classes failed with `Unsupported class file major version 65`: Spring Framework 5.3.14 cannot read Java 21 class files. Java and Spring must move together.
- **Four areas need code changes for Spring Boot 3 and later:** the security configuration (2 classes on removed APIs), the API documentation library (springfox, which does not run on Boot 3), the javax → jakarta rename (32 hand-written and 16 generated files), and Hibernate 6 query checks (28 queries).

Recommendation. Add a Safety Net of tests on the paths the current suite does not cover (PostgreSQL, MySQL, API contract, security), then upgrade to Java 21 and the current Spring Boot line in one fixed-price module. Total: **$8,100 net** after the assessment credit (section 7).

2 Scope and method

Code analyzed	83 main Java files (3,117 code lines), 18 test files (1,771 code lines), plus 16 DTO classes generated at build time from `api-docs.yml`
Shape of the app	REST API with 33 endpoints; 7 repositories with 3 interchangeable implementations each (JDBC, JPA, Spring Data JPA); HSQLDB, PostgreSQL and MySQL profiles; HTTP Basic security with database users

Build and tests	`mvn verify` on JDK 21.0.12 targeting Java 8, the project's own setting. JaCoCo coverage from the same run
Java 21 trial	Same code, compiled with `-Djava.version=21`, no other changes
Dependencies	`mvn dependency:list`: all 131 resolved artifacts with their scope
Known vulnerabilities	Each resolved artifact and version matched against the GitHub Advisory Database (reviewed advisories, snapshot of 30 Oct 2026)
Code checks	javax imports, Spring Security APIs, springfox usage, Hibernate-specific APIs, JPQL queries, deprecated API warnings (`-Xlint`), configuration properties
Not in scope	The separate Angular front end, performance testing, penetration testing, infrastructure

An advisory match means the library version is listed as affected. It does not by itself mean the application can be attacked. Section 3.1 shows our check of the critical ones against the code.

3 Findings



Critical 19 High 57 Moderate 54 Low 14 144 advisories in 29 shipped libraries

#	FINDING	SEVERITY	EVIDENCE	FIX
F1	**Framework stack is out of open-source support**	CRITICAL	Spring Boot 2.6.2, Spring Framework 5.3.14, Spring Security 5.6.1, Tomcat 9.0.56, Jackson 2.13.1. Open-source support ended for Boot 2.6 in Nov 2022, Boot 2.7 in Nov 2023 and Spring Framework 5.3 in Aug 2024	Upgrade to the current Spring Boot line (WP2)
F2	**144 known advisories in shipped libraries**	HIGH	26 compile-scope and 3 runtime-scope libraries affected. Largest: Tomcat (41), Spring MVC (17), Jackson (17), Spring Expression (8), Logback (7), PostgreSQL driver (7), SnakeYAML (7). 35 have no fix on the current line	Upgrade (WP2). No patch-only route covers the 35
F3	**Java 21 not possible on the current stack**	HIGH	Java 21 trial: 14 of 15 test classes error with `Unsupported class file major version 65` (Spring's ASM reader). JaCoCo 0.8.7 also fails to instrument JDK 21 classes (98 errors in the baseline log)	Move Java and Spring together; JaCoCo 0.8.11 or later
F4	**Spring Security configuration uses removed APIs**	HIGH	`BasicAuthenticationConfig` and `DisableSecurityConfig` extend `WebSecurityConfigurerAdapter` and use `authorizeRequests()`, `@EnableGlobalMethodSecurity` and `jdbcAuthentication()`, all removed in Spring Security 6. 34 `@PreAuthorize` rules depend on this setup	`SecurityFilterChain` beans with `authorizeHttpRequests`, `@EnableMethodSecurity` and `JdbcUserDetailsManager`; proven by security tests first (WP1)
F5	**API documentation library does not run on Boot 3**	HIGH	springfox 3.0.0 (`@EnableSwagger2`, `Docket`). It forces `ant_path_matcher` path matching and pulls Swagger 1.5 annotations into the generated DTOs, causing 20 deprecation warnings	Replace with springdoc-openapi; keep the documentation URL or redirect it
F6	**javax → jakarta namespace change**	HIGH	32 hand-written files import `javax.persistence`, `javax.validation`, `javax.transaction`, `javax.servlet` or `javax.xml.bind` (53 `javax.persistence` imports alone). 16 generated DTOs come from openapi-generator 5.2.1, which cannot emit Jakarta. `javax.sql` is part of the JDK and stays	Automated rename (OpenRewrite), then review; generator 7.x with `useJakartaEe`

#	FINDING	SEVERITY	EVIDENCE	FIX
F7	**Hibernate 5.6 → 6 query behavior**	MEDIUM	28 JPQL / `@Query` statements across the JPA and Spring Data JPA repositories. No Hibernate-specific APIs in the code, which keeps this contained	Re-run every query under Hibernate 6 against PostgreSQL in the Safety Net
F8	**Tests cover only the HSQLDB database**	MEDIUM	Test profile is fixed to `hsqldb,spring-data-jpa`. The PostgreSQL and MySQL schema and data scripts are never executed by a test	Testcontainers PostgreSQL (and MySQL if used) in WP1
F9	**Insecure defaults**	MEDIUM High in a real system	`petclinic.security.enable=false`, so every endpoint is open by default. Seed data creates user `admin` with password `{noop}admin` (plain text). Database passwords sit in plain properties files	Security on by default, BCrypt-hashed seed users, secrets from the environment
F10	**Abandoned library in the JDBC layer**	MEDIUM	`spring-data-jdbc-core` 1.2.1.RELEASE (built Nov 2015, retired Spring Data JDBC Extensions project), used by one class: `JdbcPetVisitExtractor`	Replace with a small `ResultSetExtractor` in the code base
F11	**Build tooling is dated**	LOW	CI builds on Java 11 with `actions/checkout@v2`; jib-maven-plugin 1.3.0; javac warns source/target 8 are obsolete; `mysql-connector-java` moved to `com.mysql:mysql-connector-j`	Update plugins and CI to Java 21 in WP2
F12	**Small hygiene items**	LOW	Spring Boot 1.x properties left in test config (`security.ignored`, `basic.authentication.enabled`); 5 MapStruct unmapped-property warnings; `CallMonitoringAspect` has 0% coverage	Clean up during WP2

3.1 Critical advisories checked against the code

ADVISORY	LIBRARY	APPLIES HERE?	WHY
CVE-2022-22965 (Spring4Shell)	spring-beans 5.3.14	NO in the known form	Exploit needs WAR deployment on Tomcat; this app runs as an executable jar
CVE-2016-1000027	spring-web 5.3.14	NO	Only affects HttpInvoker remoting, which is not used
CVE-2025-24813	Tomcat 9.0.56	NO by default	Needs a write-enabled default servlet (partial PUT), which is off by default
CVE-2026-43512, CVE-2026-65905, CVE-2026-68525	Tomcat 9.0.56	NO	Affect Tomcat's own DIGEST / FORM login; the app authenticates through Spring Security

ADVISORY	LIBRARY	APPLIES HERE?	WHY
CVE-2022-22978	spring-security-core 5.6.1	NO	Needs `RegexRequestMatcher`; the app uses `anyRequest()`
CVE-2023-34034, CVE-2024-38821	Spring Security 5.6.1	NO	WebFlux only; the app is Spring MVC
CVE-2023-20860	spring-webmvc 5.3.14	NO	Needs `mvcMatchers` with `**`; not used
CVE-2023-20873	Boot actuator 2.6.2	NO	Cloud Foundry deployments only
CVE-2024-1597	PostgreSQL driver 42.3.1	NO	Needs `preferQueryMode=simple`; not set
CVE-2022-41853	HSQLDB 2.5.2	LOW	Needs untrusted input in SQL; all queries are parameterized. HSQLDB should not be a production database anyway
CVE-2022-31692, CVE-2026-22732, CVE-2026-47884, CVE-2026-65182, CVE-2026-43515, CVE-2026-41293	Spring Security, Spring MVC, Tomcat	OPEN not confirmed either way	Need deeper review; the upgrade removes all six

High-severity advisories that **do apply**: Jackson denial-of-service issues (for example CVE-2022-42003, CVE-2022-42004, CVE-2020-36518), because every endpoint parses JSON from the caller; Tomcat request-smuggling and multipart issues (for example CVE-2026-24880, CVE-2025-48988), because Tomcat is the public HTTP server.

4 Upgrade path

Target: **Java 21 and the current supported Spring Boot line** (Spring Boot 4.1 at the time of writing), reached through Spring Boot 3.5 so each step stays small and testable. If a client must stop at Spring Boot 3.5, the path ends at step 4.

`Safety Net`

1. Tests first (WP1)

PostgreSQL integration tests, API contract tests for all 33 endpoints with security on, security rule tests, CI on a Java 21 runner. Behavior is recorded before anything changes.

`Boot 2.7 · Java 8`

2. Last 2.x release

Clears part of the advisories and surfaces deprecation warnings that point at what Boot 3 removes. Security is rewritten to `SecurityFilterChain` here, because Spring Security 5.7/5.8 supports both styles.

`Boot 2.7`

3. Replace dead libraries

springfox to springdoc; the 2015 JDBC extension library to a local extractor. The path-matcher workaround goes.

`Boot 3.5 · Java 21`

4. Jakarta and Java 21

javax → jakarta rename (automated, then reviewed), openapi-generator 7.x with Jakarta, Hibernate 6 query fixes, `mysql-connector-j`, JaCoCo and plugin upgrades.

`Boot 4.1 · Java 21`

5. Current line

Spring Framework 7, Hibernate 7, Jackson 3. Full test suite and advisory scan run again.

`Hand-over`

6. Hand-over

No known critical or high advisories in shipped libraries, all tests green on Java 21, change log and rollback notes.

5 Risks and how we control them

RISK	CONTROL
Security rules behave differently after the rewrite	Security tests for every `@PreAuthorize` rule and for unauthenticated access are written in WP1 and must pass before and after
Hibernate 6/7 changes query results	All 28 queries run against real PostgreSQL in Testcontainers and are compared to the WP1 baseline
API responses change shape (Jackson 3, generated DTOs)	API contract tests record the current JSON for each endpoint; any difference fails the build
PostgreSQL / MySQL paths break unseen	Untested today (F8); WP1 adds them before any upgrade
API documentation URL changes	springdoc configured to the same path, or a redirect agreed before WP2
New advisories appear after hand-over	Keep-Current Retainer (section 7)

6 Effort estimate

WORK PACKAGE	MAIN ITEMS	ESTIMATE
WP1 Safety Net	Testcontainers PostgreSQL tests, 33-endpoint contract tests, security tests, CI on Java 21	4–5 days
WP2 Upgrade & Migration	Steps 2–6: security rewrite (2 classes, 34 rules), springdoc, JDBC library swap, Jakarta (48 files incl. generated), 28 queries, plugins, CI, Boot 3.5 then 4.1	8–10 days

These are Pashyaa estimates based on the counts above, not measurements. The existing 171 tests and 88% coverage reduce both; a code base with weak tests would cost more.

7 Fixed-price quote

All prices in USD. Fixed scope, fixed price.

STEP	WHAT YOU GET	PRICE
Modernization Assessment	This report, the evidence pack and a walkthrough call	$900 credited in full
Safety Net	WP1, delivered as a pull request with a test report	$2,500
Upgrade & Migration Module	WP2, delivered as reviewed pull requests, one per step, with a hand-over report	$6,500
Total after credit	**Safety Net + Upgrade, less the $900 assessment**	**$8,100**
Keep-Current Retainer optional	Monthly advisory scan, patch and minor upgrades, test run report, Spring Boot minor upgrade every six months	$500 / month

Done means

All existing and new tests pass on Java 21. The advisory scan shows no known critical or high advisory in shipped libraries at hand-over. API contract tests show no unintended change.

Assumptions

Read access to the repository and a CI runner; production-like sample data or permission to use the seed data; one named contact. The front end, data migration and infrastructure are out of scope and quoted separately if needed.

Evidence pack (build logs, Java 21 trial log, dependency list, coverage data, the full list of advisory matches (144 in shipped libraries, plus 4 in test-only libraries) and the matching script) is available on request.

Pashyaa Technologies Pvt. Ltd., Pune, India · Java and Spring Boot modernization at a fixed price

pashyaa.com · info@pashyaa.com