MPL Lab 11
Mayur Jaiswal
D15B
24

Aim: To use google Lighthouse PWA Analysis Tool to test the PWA functioning.

Theory: Google Lighthouse is an open-source tool built into Chrome DevTools. It provides detailed reports on performance, accessibility, best practices, SEO, and PWA compliance. It simulates loading conditions and gives scores, along with actionable recommendations.

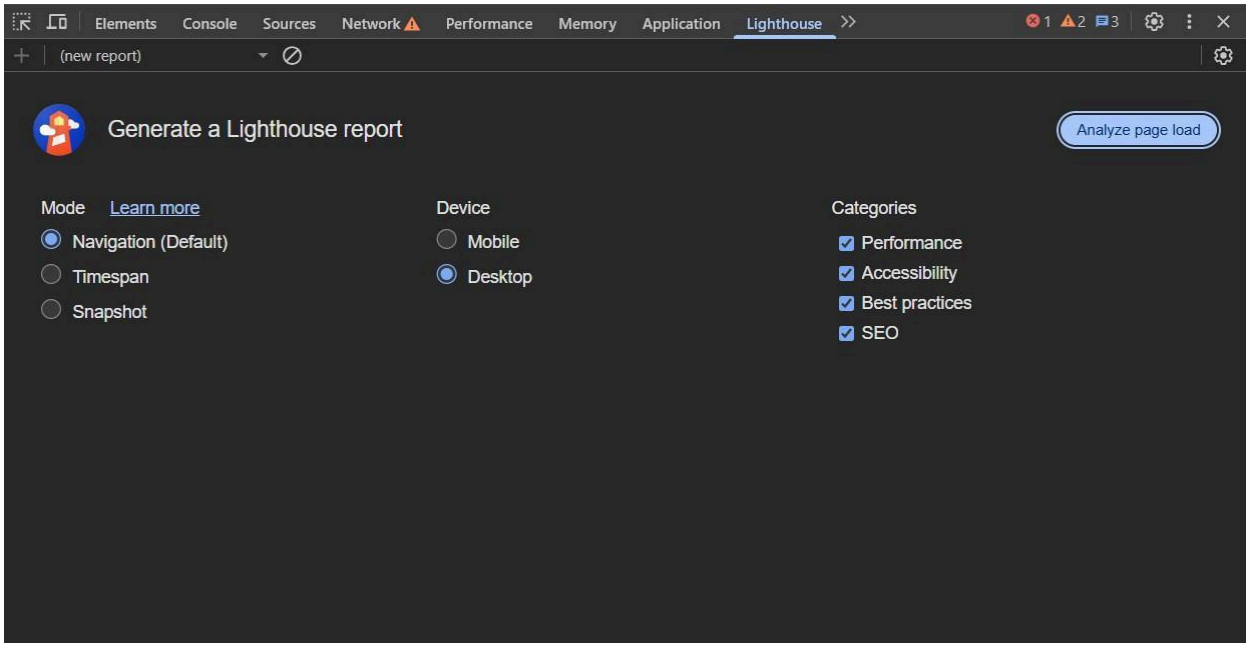
It is crucial for optimizing the user experience, especially for mobile users and slow network conditions.

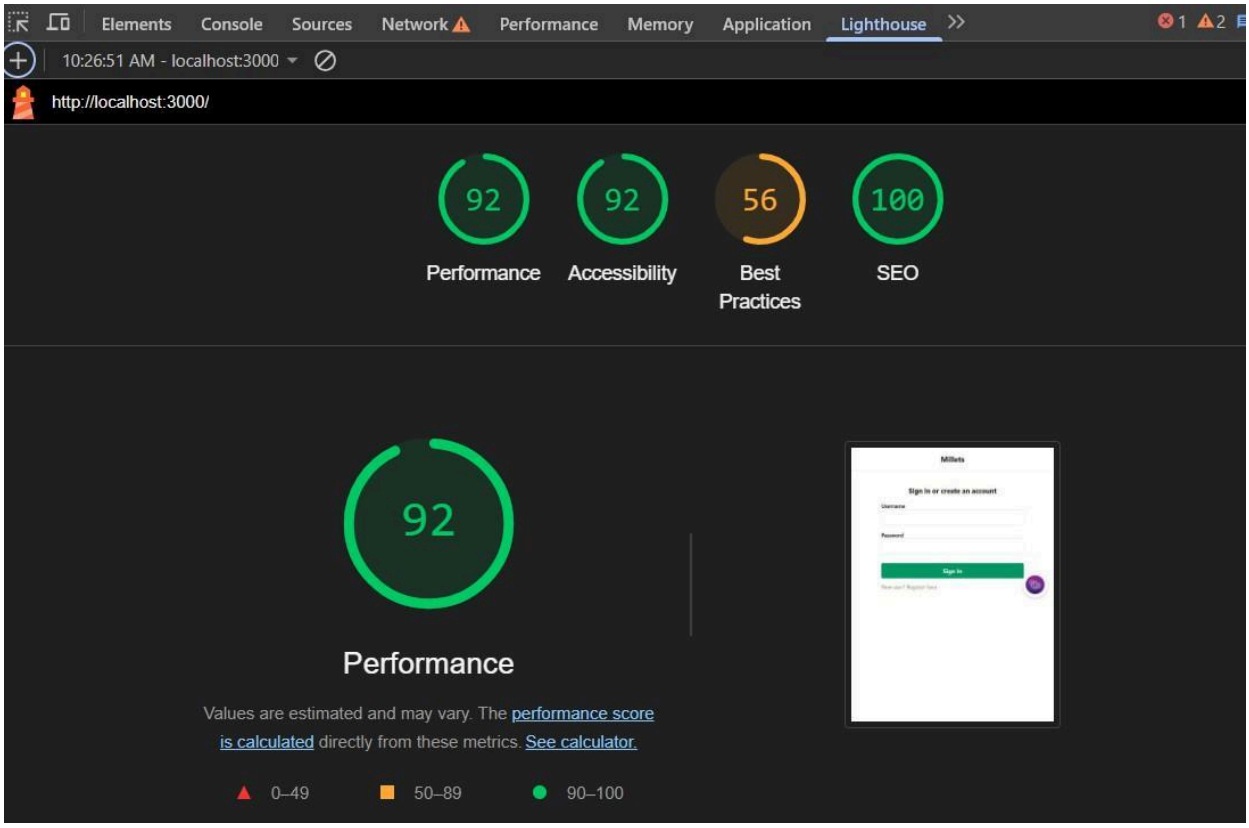
Steps to Perform:

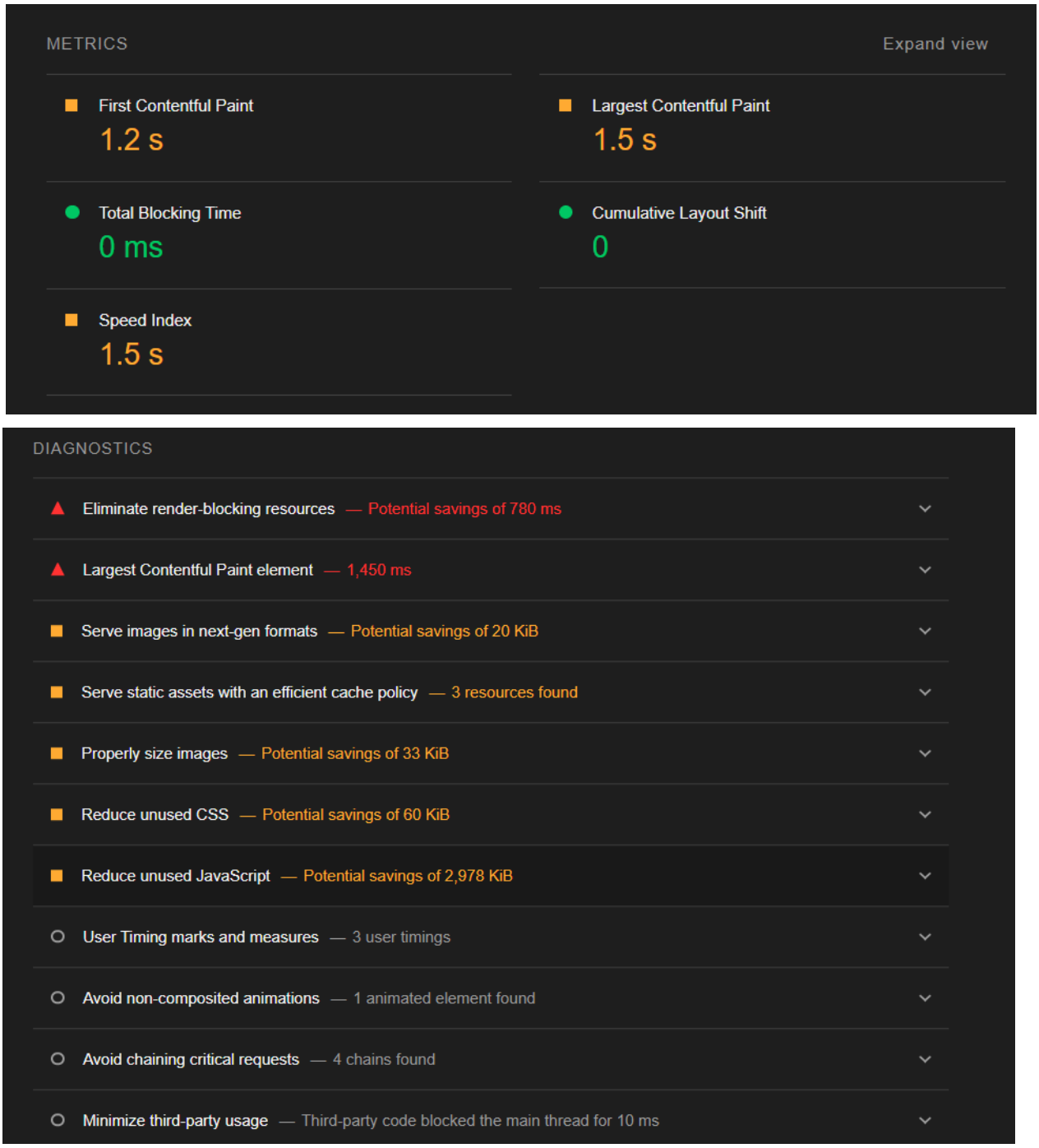
1. Open Your App in Chrome: Ensure your app is running on HTTPS or localhost.
2. Access Lighthouse:
    - Right-click → Inspect → DevTools.
    - Navigate to the Lighthouse tab.
    - Choose the categories: Performance, PWA, Accessibility, etc.
    - Select device type: Mobile/Desktop.
3. Generate Report:
    - Click “Generate Report”.
    - Review your scores and suggestions.
4. Fix Issues:
    - Implement suggestions like image compression, lazy loading, better caching, improved service worker registration, manifest improvements, etc.
5. Re-test After Changes:
   Re-run Lighthouse to track improvements.

Key Features:

- Progressive Web App Audit: Verifies installability, service worker, and offline readiness.
- Performance Optimization: Identifies loading bottlenecks.
- Accessibility Check: Ensures app is usable for everyone.
- Best Practices and SEO: Improves overall quality and searchability.


Elements
Console
Sources
Network
Performance
Memory
Application
Lighthouse
(new report)
Generate a Lighthouse report
Analyze page load
Mode
Learn more
Navigation (Default)
Timespan
Snapshot
Device
Mobile
Desktop
Categories
Performance
Accessibility
Best practices
SEO


Elements
Console
Sources
Network
Performance
Memory
Application
Lighthouse
10:26:51 AM - localhost:3000
http://localhost:3000/
92
Performance
92
Accessibility
56
Best Practices
100
SEO
92
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49
50–89
90–100


METRICS
Expand view
First Contentful Paint
1.2 s
Largest Contentful Paint
1.5 s
Total Blocking Time
0 ms
Cumulative Layout Shift
0
Speed Index
1.5 s
DIAGNOSTICS
Eliminate render-blocking resources — Potential savings of 780 ms
Largest Contentful Paint element — 1,450 ms
Serve images in next-gen formats — Potential savings of 20 KiB
Serve static assets with an efficient cache policy — 3 resources found
Properly size images — Potential savings of 33 KiB
Reduce unused CSS — Potential savings of 60 KiB
Reduce unused JavaScript — Potential savings of 2,978 KiB
User Timing marks and measures — 3 user timings
Avoid non-composited animations — 1 animated element found
Avoid chaining critical requests — 4 chains found
Minimize third-party usage — Third-party code blocked the main thread for 10 ms