**Project Development Phase**

**Model Performance Test**

|  |  |
| --- | --- |
| Date | 1st, April 2025 |
| Team ID | SWTID1742493942 |
| Project Name | Connectify - Social Media App |
| Maximum Marks | 4 Marks |

**6. Performance Testing**

**6.1 Objectives, Types, Metrics, Tools, Scenarios, Optimization & Reporting**

Performance testing ensures that the social media platform remains **fast, reliable, scalable**, and can handle real-world user interactions smoothly across frontend and backend systems. The key objectives and elements of performance testing for our project are detailed below:

**Objectives of Performance Testing**

**Ensure Fast and Responsive User Experience**

* **Feed load time** ≤ 2 seconds
* **Media (image/video) load time** ≤ 3 seconds
* **Reminder panel toggle time** ≤ 1 second
* **Notification panel** and **message inbox** must respond quickly under stress.

**Handle Concurrent Users and Real-Time Actions**

Simulate high-concurrency user scenarios like:

 500 users logging in simultaneously

 300 users searching concurrently

 500 users creating posts or reminders

**Backend/API Performance**

API **response time** ≤ 200 ms for key endpoints for login, post upload, fetch feed, send message)

Use tools like **Postman** for load and performance testing

 200 users updating or editing their reminder

 500 users interacting with the feed

**Database Performance**

* **MongoDB query response time** ≤ 300 ms even under 1000+ concurrent users
* Optimize using:
  + **Indexing** on frequently queried fields (e.g., userId, postId, createdAt)
  + **Redis caching** for frequently accessed data (like popular posts or notifications)
  + **Efficient schema design** to minimize joins and improve lookup speed

**Frontend and Cross-Platform Responsiveness**

* Page load time ≤ 1 second (including feed, profile, reminder panel)
* **Test across:**
  + Android and iOS devices
  + Desktop (Chrome, Firefox, Edge)
  + Simulated 3G/4G/5G networks
* Tools: **Chrome DevTools**, **BrowserStack**

**Resource & Browser Performance**

* CPU/Memory usage ≤ 80%
* No browser crashes or lag under load
* Smooth tab switching and app navigation

**Efficient LocalStorage Usage**

* Read/Write time ≤ 50ms
* Total storage usage < 5MB (browser limit)
* No data loss on refresh or tab close

**Scalability and Fault Tolerance**

* Test System Scaling with Users and Content Growth
* Simulate Heavy Load Conditions
* Optimize for LocalStorage Limitations
* Handle Fault Tolerance with LocalStorage.

**Types of Performance Testing**

| Type | Description | Example |
| --- | --- | --- |
| Load Testing | Test Local Storage limits | 100+ reminders, 100+ posts |
| Stress Testing | Beyond normal limits | Exceeding 5MB storage with large dummy posts |
| UI Responsiveness | Check frontend response under heavy interaction | Measure rendering delay while scrolling long feed |
| Spike Testing | Sudden burst of user actions | 50 likes and 20 reminders created in 2 minutes |
| Device Testing | Performance across different screen sizes and devices | Test on desktop, tablet, and mobile via DevTools |
| Volume Testing | Handling large amount of user-generated data in LocalStorage | Store thousands of likes, posts, and user actions |
| Latency Testing | Delay between user action and UI update | Check delay between clicking "Like" and UI update |
| Concurrent Testing | Simulate multiple tabs/sessions doing actions simultaneously | Add/reminders in Tab 1 and verify sync in Tab 2 |

**Key Performance Metrics**

| Metric | Ideal Target |
| --- | --- |
| |  |  | | --- | --- | | Page Load Time |  | | |  |  | | --- | --- | |  | ≤ 1.5 seconds | |
| |  |  | | --- | --- | | Reminder Setup Time |  | | |  |  | | --- | --- | |  | ≤ 500 ms | |
| |  |  | | --- | --- | | Post Creation Time |  | | |  |  | | --- | --- | |  | ≤ 1 second | |
| |  |  | | --- | --- | | LocalStorage Read/Write |  | | |  |  | | --- | --- | |  | ≤ 50 ms | |
| |  |  | | --- | --- | | CPU/Memory Usage (browser) |  | | |  |  | | --- | --- | |  | ≤ 80% | |
| |  |  | | --- | --- | | Storage Capacity Used |  | | |  |  | | --- | --- | |  | ≤ 5MB | |
| |  | | --- | | UI Response Time | | |  |  | | --- | --- | |  | ≤ 300 ms | |
| |  |  | | --- | --- | | Data Retention on Reload |  | | |  |  | | --- | --- | |  | 100% (no loss) | |
| |  |  | | --- | --- | | Tab Switching Performance |  | | |  |  | | --- | --- | |  | Smooth (≤ 500ms delay) | |

**Performance Testing Tools**

| Area | Tools |
| --- | --- |
| |  |  | | --- | --- | | UI Load and Speed |  | | |  |  | | --- | --- | |  | Chrome DevTools, Lighthouse | |
| |  |  | | --- | --- | | Storage Behavior |  | | |  |  | | --- | --- | |  | Application Tab in DevTools (LocalStorage) | |
| |  |  | | --- | --- | | Network & CPU Monitoring |  | | |  |  | | --- | --- | |  | DevTools → Performance + Memory tabs | |
| |  |  | | --- | --- | | Rendering Optimization |  | | |  |  | | --- | --- | |  | React Profiler | |
| |  |  | | --- | --- | | Browser Compatibility |  | | |  |  | | --- | --- | |  | Manual testing on Edge, Firefox, Chrome | |
| |  |  | | --- | --- | | Error Checking |  | | |  |  | | --- | --- | |  | Console log inspection | |

**Test Scenarios**

**Reminder Functionality**

* Create 20+ reminders quickly
* Verify display on left panel instantly
* Refresh browser → Check persistence

**Post/Like/Delete Actions**

* Simulate multiple actions in rapid succession
* Post → Like → Comment → Delete → Undo

**Multiple Tabs**

* Open app in multiple tabs
* Add a reminder in one → Check if visible in another after reload

**Device Performance**

* Test on desktop & mobile
* Add content and scroll through feed

**Storage Limit Test**

* Fill up LocalStorage with dummy data
* Check performance and system behavior (graceful degradation)

**Page Navigation**

* Switch between Home, Profile, Reminder quickly
* Confirm UI is stable and fast

**Test Reporting and Analysis**

**Monitor Response Times:** Track the response times for user actions like liking posts, adding reminders, and fetching data from LocalStorage (min, max, avg).

**Analyze Errors:** Identify common errors such as timeouts or LocalStorage limit issues (e.g., exceeding 5MB storage).

**Track Resource Usage:** Monitor CPU, memory, and storage usage, especially when interacting with a large number of posts or reminders.

**Simulate User Behavior:** Assess the app's performance during common actions like adding posts, editing reminders, or scrolling through feeds.

**Identify Bottlenecks:** Pinpoint delays in data fetching or saving due to LocalStorage limitations or inefficient data handling.

**Provide Recommendations:** Suggest improvements like optimizing data storage, reducing memory usage, and enhancing UI rendering for better performance.