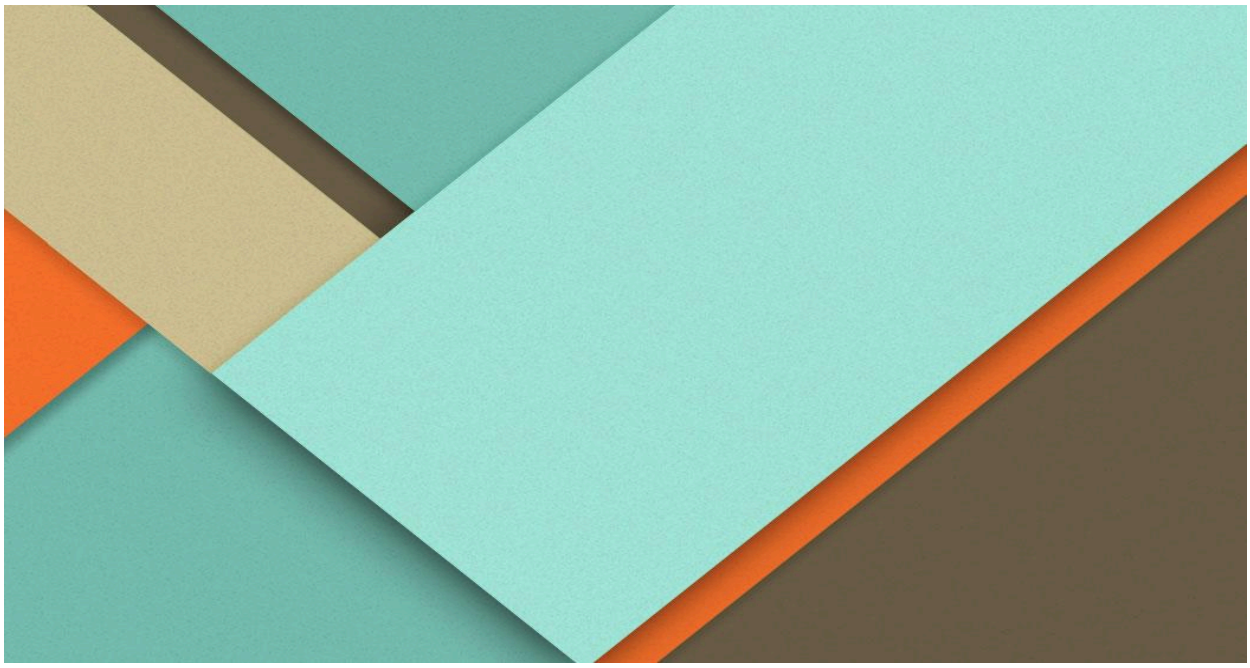


Software Testing Assignment-2

23.03.2025

Mahmut Esad Erman, Necmettin Bera Çalık, Nicolai Glock, Ahmad Alhelal

Requirements

Functional Requirements:

1. Users should be able to create, edit, and delete individual tasks with ease.
2. Each task includes a title, description, and contextual details to clarify its purpose.
3. Each task should have an associated status (e.g., pending, in progress, completed) to provide clear workflow feedback.
4. Users can attach dates to tasks, ensuring they are aware of deadlines and upcoming priorities.
5. Provide timely notifications and reminders that help keep the user aware of time-sensitive items without causing excessive disruption.
6. The system should be suitable for small teams
7. Users should be able to assign or share tasks with others
8. There should be a login and register system.

Non-Functional Requirements:

9. The software is web-based.

Quality Models

1. **Usability**
    - **Definition:** The system should be easy to use, intuitive, and accessible.
    - **Sub-characteristics:**
        - **Learnability:** Users should quickly understand how to create and manage tasks.
        - **Operability:** The interface should be simple and responsive.

        - **User Interface Aesthetics:** A clean, distraction-free UI.
    - **Quality Measures:**
        - User satisfaction surveys (e.g., target: 80% positive feedback).
        - Average time taken to complete a task (target: <10 seconds).
2. **Reliability**
    - **Definition:** The system should function without crashes or data loss.
    - **Sub-characteristics:**
        - **Fault Tolerance:** The system should handle failures gracefully.
        - **Availability:** The system should be accessible with minimal downtime.
        - **Recoverability:** Users should not lose tasks due to unexpected failures.
    - **Quality Measures:**
        - System uptime (target: 99.9%).
        - Task data recovery rate (target: 100% after unexpected shutdowns).
3. **Performance Efficiency**
    - **Definition:** The system should respond quickly and not consume excessive resources.
    - **Sub-characteristics:**
        - **Time Behavior:** Task updates should be near-instantaneous.
        - **Resource Utilization:** The application should run efficiently on various devices.

- **Quality Measures:**
  - Response time for task updates (target: <1 second).
  - CPU and memory usage benchmarks.

4. **Maintainability**
   - **Definition:** The codebase should be structured for easy updates and fixes.
   - **Sub-characteristics:**
     - **Modularity:** Clear separation between UI, logic, and database layers.
     - **Testability:** Automated tests should be easy to implement.
   - **Quality Measures:**
     - Code coverage in unit tests (target: 80%).
     - Number of critical bugs in new releases (target: <2 per update).

---

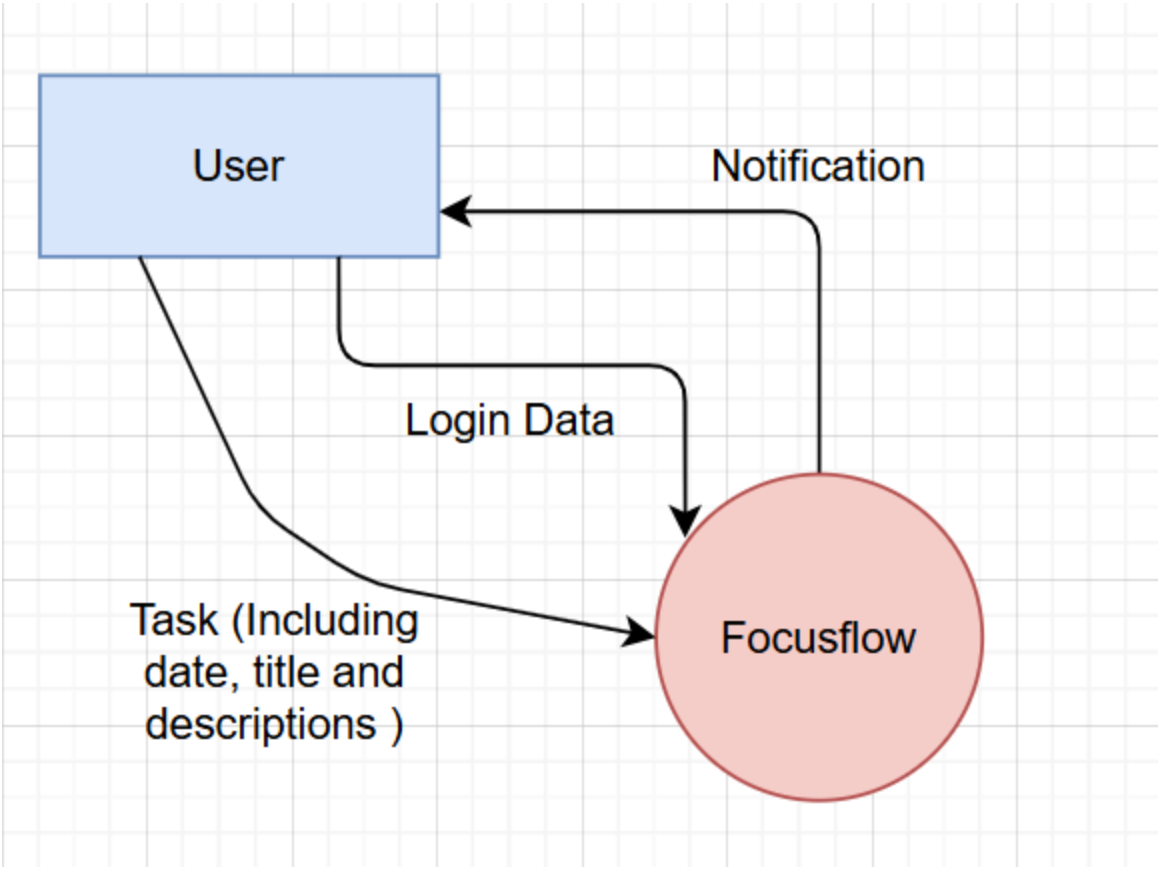
2. Testability Measures (Ensuring Quality)

To improve **testability**, the following practices should be implemented:

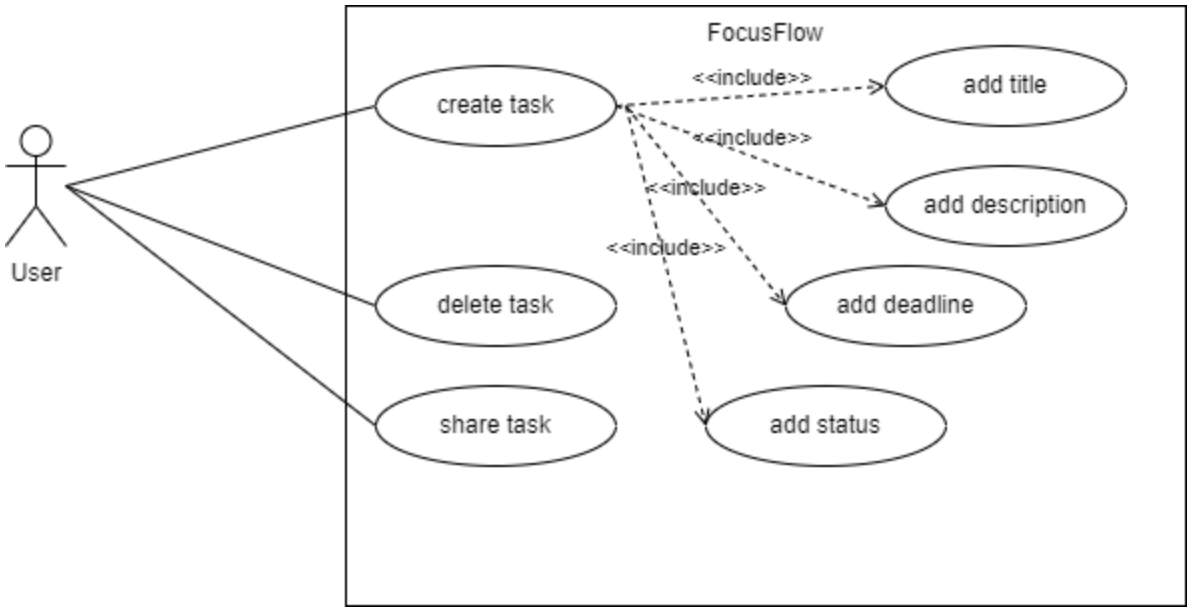
- **Automated Testing:** Unit, integration, and UI tests.
- **Mocking & Stubbing:** Simulate interactions with external services.
- **Logging & Monitoring:** Track performance issues and errors.
- **Version Control & CI/CD Pipelines:** Ensure smooth deployments.

System Context & Use Cases

I. Context Diagram



II. Use Case Diagram


FocusFlow
User
create task
delete task
share task
add title
add description
add deadline
add status
<<include>>
<<include>>
<<include>>
<<include>>