**Avantaje şi dezavantaje TDD**

Dezvoltarea condusă de testare (TDD) este o metodă în dezvoltarea software-ului în care testele sunt create înainte de a se scrie codul efectiv. Această abordare implică un proces ciclic: se începe cu elaborarea testelor, apoi se scrie codul pentru a trece aceste teste și în final se optimizează și se îmbunătățește codul existent.

**Avantaje**

1. **Îmbunătățirea calității codului:** TDD încurajează redactarea unui cod ordonat, modular și ușor de întreținut. Prin scrierea testelor înainte de cod, acesta este conceput pentru a îndeplini cerințe precise, ceea ce duce la implementări de calitate superioară.
2. **Debugging mai rapid:** Identificând erorile devreme în procesul de dezvoltare, TDD reduce riscul de apariție a unor erori majore în cod. În consecință, procesul de depanare devine mai simplu, având în vedere că testele localizează precis sursa erorilor.
3. **Claritate în design:** TDD încurajează o înțelegere precisă a cerințelor înainte de a începe implementarea. Acesta facilitează definirea clară a comportamentului pe care se așteaptă să-l aibă codul, contribuind la un proces de dezvoltare mai concentrat și eficient.
4. **Testare de regresie:** TDD asigură că ajustările sau îmbunătățirile nu afectează funcționalitățile deja existente. Prin rularea frecventă a setului de teste, dezvoltatorii sunt siguri că pot efectua modificările necesare fără grija de a afecta alte componente ale sistemului.
5. **Susține refactorizarea:** Pe măsură ce codul se dezvoltă, dezvoltatorii pot efectua refactorizări fără griji, având încredere că testele deja existente vor garanta păstrarea comportamentului dorit al codului.

**Dezavantaje**

1. **Adaptarea inițială la TDD:** Implementarea TDD necesită ca dezvoltatorii să se familiarizeze cu noi practici și să se acomodeze să scrie teste înainte de a crea cod. Această perioadă de adaptare poate încetini ritmul inițial al dezvoltării.
2. **Consum considerabil de timp:** Se poate considera că adăugarea pasului de scriere a testelor înainte de cod poate să prelungească inițial procesul de dezvoltare, făcându-l mai consumator de timp. Această situație poate fi percepută ca un dezavantaj, în special atunci când există termene strânse.
3. **Menținerea testelor:** Menținerea suitei de teste necesită atenție similară cu cea acordată bazei de cod. Odată cu progresul software-ului, testele pot necesita ajustări sau revizuiri pentru a se adapta la noile caracteristici sau schimbările de cerințeTop of Form.
4. **Legacy şi testarea TDD:** Integrarea TDD în sistemele legacy sau în proiectele cu cod extins deja existent poate fi problematică. Dezvoltarea testelor pentru codul legacy poate fi dificilă sau chiar impractică în anumite situații.
5. **Limitări ale TDD:** Deținerea testelor nu asigură un sistem fără erori. Deși TDD ajută la identificarea multor probleme în stadiile incipiente, nu acoperă fiecare posibil scenariu sau nu garantează un cod perfect.