

Proces complet de debugging - manual și automat

Versiune document: 2 iulie 2026

Proces complet de debugging - manual și automat

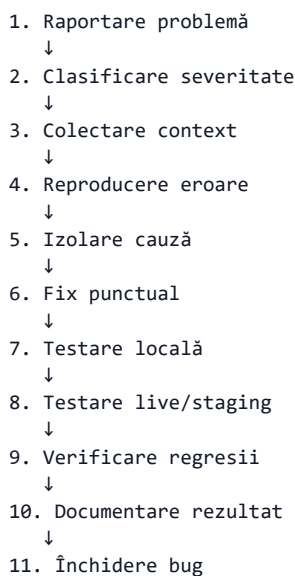
Versiune document: 2 iulie 2026 Aplicație: FiscontExpert Consult Scop: metodologie completă de identificare, reproducere, testare, reparare și validare a erorilor.

1. Obiectiv

Procesul de debugging trebuie să răspundă clar la întrebările:

- Ce s-a stricat?
- Unde se reproduce?
- Cine este afectat?
- Este eroare locală, live, de cod, de date, de hosting sau de configurare?
- Cum se testează?
- Ce rezultat se așteaptă?
- Ce rezultat s-a obținut?
- Ce fix s-a aplicat?
- Ce trebuie retestat?

2. Schema generală debugging



3. Clasificare severitate

Severitate	Descriere	Exemple	Timp reacție recomandat
S0 - Blocant	Site-ul sau adminul nu funcționează deloc.	503 live, login imposibil, DB picată.	Imediat
S1 - Critic	Funcție majoră nefuncțională.	salvare pagini nu merge, documente inaccesibile, portal blocat.	În aceeași zi
S2 - Major	Funcție importantă afectată, workaround posibil.	layout rupt pe mobil, raport GDPR incomplet.	1-2 zile
S3 - Minor	Problemă vizuală sau text.	alinieri, typo, icon greșit.	Planificat
S4 - Îmbunătățire	Nu este bug, ci optimizare.	export PDF nou, automatizare nouă.	Backlog

4. Ce se testează

4.1. Site public

Zonă	Ce se testează	Rezultat așteptat
Acasă	Header, hero, carduri, noutăți, footer	Arată corect și responsive
Despre noi	Text, iconuri, buton ofertă	Conținut aliniat și complet
Premii	Galerie, poziționare text, imagini	Text sub imaginea corectă
Servicii	Acordeon, butoane, griuri	Design identic/consistent
Noutăți	Carduri, fonturi, imagini, linkuri	Font și spațiere corecte
Cariere	Banner, conținut, joburi	Fără margini/culori greșite
Contact	Formular, date firmă, hartă	Formular și hartă poziționate corect
Footer	Linkuri, comunitate, distribuie	Linkuri corecte, iconuri simetrice
Limbi	RO/EN/FR/IT	Textele se schimbă complet
Cookie	Banner/buton și preferințe	Necesare blocate, opționale editabile

4.2. Administrare

Zonă	Ce se testează	Rezultat așteptat
Login admin	Autentificare, 2FA	Intră corect în admin
Sidebar	Scroll independent	Nu derulează pagina principală accidental
Panou general	Statistici, mentenanță	Date afișate corect
Pagini site	Salvare conținut	Modificarea se vede public
Template-uri	Selectare/aplicare	Template-ul activ se schimbă
SEO/Ads	Salvare setări	Setările persistă
GDPR/DPIA/GAP	Rulare raport	Raport pe https://www.fiscontexpert.ro
Email marketing	Campanie test	Email trimis sau eroare clară
Chat intern	Setări și răspuns	Widgetul respectă setările

Zonă	Ce se testează	Rezultat așteptat
Plăți	Activare/dezactivare	Meniul Plătește apare/dispare
Utilizatori	creare/assignare	Userul vede firmele corecte
Companii	status/billing	Datele persistă
Integrări API	salvare credentiale	Date criptate și status corect
Documente	upload/download	Acces securizat
Ștergeri GDPR	cerere/retenție	Nu se șterg date obligatorii legal
Backup/restore	snapshot/căutare	Backup căutabil

4.3. Portal client

Zonă	Ce se testează	Rezultat așteptat
Login client	Acces portal	Clientul intră în cont
Firme asignate	Listare firme	Vede doar firmele lui
Documente	Upload/download	Documente accesibile corect
Analiză	generare raport	Raport creat dacă există date
Avertisment plată	mesaj roșu	Apare doar când este activat
Contul meu	parolă, email, 2FA	Setările se schimbă corect
Ștergere cont	cerere GDPR	Se aplică retenție dacă există date legale

4.4. Backend/API

API	Ce se testează	Rezultat așteptat
/api/anaf/company	CUI numeric	Returnează firmă sau eroare controlată
/api/chat/message	mesaj vizitator	Creează sesiune și mesaj
/api/documente/[id]	acces document	Servește doar dacă userul are drept
/api/easter-egg/appearance	aparitie simbol	Respectă limitele lunare
/api/easter-egg/claim	revendicare	Generează cod și inventar

4.5. Hosting/deploy

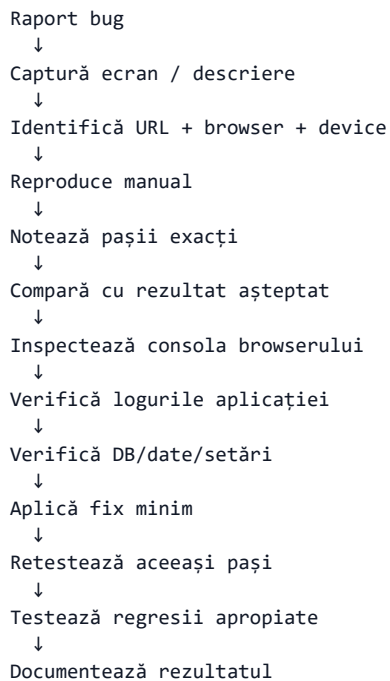
Zonă	Ce se testează	Rezultat așteptat
Node app	cPanel app pornită	Nu apare 503
.next/static	asseturi disponibile	CSS/JS se încarcă
DB	conexiune Prisma	Query funcțional
env	variabile corecte	URL, DB, secret corecte
runtime log	erori	Fără erori critice

5. Variantă 1 - Debugging manual

Debuggingul manual este potrivit pentru:

- probleme vizuale;
- probleme raportate de utilizator;
- testare cross-browser;
- fluxuri complexe;
- verificare conținut;
- GDPR și UX.

5.1. Schema debugging manual



5.2. Fișă manuală de bug

ID bug:
Data:
Raportat de:
Mediu: local / live / staging
URL:
Browser:
Device:
Rezoluție:
User/rol:
Severitate:

Descriere:

Pași reproducere:

- 1.
- 2.
- 3.

Rezultat așteptat:

Rezultat obținut:

Capturi / video:

Console errors:

Loguri server:

Cauză probabilă:

Fix aplicat:

Fișiere modificate:

Retestare:

Rezultat final:

Status: deschis / în lucru / rezolvat / amânat / nu se reproduce

5.3. Testare manuală site public

Pași:

- 1 Deschide homepage.
- 2 Verifică headerul.
- 3 Verifică meniul.
- 4 Schimbă limbile.
- 5 Verifică hero.
- 6 Derulează până jos.
- 7 Verifică footerul.
- 8 Deschide fiecare pagină publică.
- 9 Verifică formularele.
- 10 Verifică butonul cookie.
- 11 Verifică chatul.
- 12 Verifică mobil/tabletă.

Rezultat așteptat:

- nicio suprapunere;
- fonturi corecte;
- imagini corecte;
- steaguri corecte;
- texte traduse;
- linkuri funcționale;
- responsive corect.

5.4. Testare manuală admin

Pași:

- 1 Login admin.
- 2 Verifică sidebar.
- 3 Intră în fiecare meniu.
- 4 Salvează o setare minoră.
- 5 Revino și verifică persistarea.
- 6 Editează o pagină.
- 7 Verifică public modificarea.

- 8 Rulează raport GDPR.
- 9 Verifică chat/plăți/email.
- 10 Verifică documente/utilizatori/firme.

Rezultat așteptat:

- fără pagini fără CSS;
- fără erori 500;
- fără formulare care nu salvează;
- fără meniuri suprapuse;
- fără scroll blocat.

5.5. Testare manuală portal client

Pași:

- 1 Login client.
- 2 Verifică firmele asignate.
- 3 Verifică documentele.
- 4 Încarcă fișier test.
- 5 Generează analiză dacă există date.
- 6 Verifică pagina Contul meu.
- 7 Activează/dezactivează 2FA.
- 8 Testează resetare parolă.

Rezultat așteptat:

- clientul nu vede datele altui client;
- documentele sunt protejate;
- analiza pornește doar cu date;
- mesajele sunt clare.

5.6. Testare manuală responsive

Viewporturi recomandate:

Device	Rezoluție
Mobile mic	360x640
Mobile modern	390x844
Mobile mare	414x896
Tabletă	768x1024
Laptop	1366x768
Desktop	1920x1080
Kiosk	1920x1080 sau rezoluția reală

Browsere:

- Chrome;
- Edge;
- Firefox;
- Waterfox;
- Opera.

Rezultat așteptat:

- meniurile nu intră sub imagini;
- butoanele nu ies din container;
- tabelele se adaptează;
- adminul are scroll independent;
- portalul este utilizabil pe mobil.

5.7. Rezultate debugging manual

Rezultatul se notează într-un tabel:

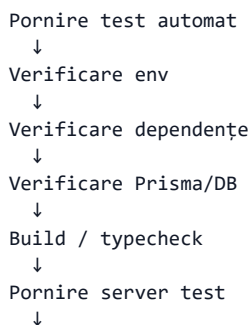
Test	Mediu	Rezultat așteptat	Rezultat obținut	Status	Observații
Homepage desktop	live	Layout corect	Layout corect	PASS	-
Admin pagini	live	Salvează text	Nu salvează	FAIL	verificat action/update
Portal documente	local	upload OK	upload OK	PASS	-

6. Variantă 2 - Debugging automat

Debuggingul automat este potrivit pentru:

- verificări repetitive;
- rute critice;
- build;
- lint;
- conexiune DB;
- API-uri;
- smoke test live;
- regresii după deploy.

6.1. Schema debugging automat



```

Smoke test rute publice
↓
Smoke test auth/admin/portal
↓
Smoke test API
↓
Verificare SEO/GDPR basic
↓
Verificare responsive cu browser automation
↓
Generare raport PASS/FAIL
↓
Oprire server test

```

6.2. Nivele de test automat

Nivel	Ce verifică	Unelte recomandate
Static	TypeScript, lint, formatare	npm run build, npm run lint
DB	Prisma Client, conexiune, schema	npm run db:generate, query Prisma
API	endpointuri interne	fetch, Playwright API, curl
UI smoke	rute publice/admin/portal	Playwright
Responsive	viewporturi	Playwright screenshots
SEO/GDPR basic	meta, canonical, lang, headers	script custom
Deploy	live healthcheck	script custom/curl

7. Checklist automat local

7.1. Comenzi de bază

```

npm install
npm run db:generate
npm run build

```

Opțional:

```
npm run lint
```

7.2. Test conexiune DB

```

node -e "const {PrismaClient}=require('@prisma/client'); const p=new PrismaClient(); p.
siteSetting.count().then(c=>{console.log('siteSetting=',c); return p.$disconnect()}).catch(
async e=>{console.error(e); await p.$disconnect(); process.exit(1)})"

```

Rezultat așteptat:

```
siteSetting= număr
```

7.3. Verificare env

```

node -e "console.log({NODE_ENV:process.env.NODE_ENV,NEXT_PUBLIC_SITE_URL:process.env.
NEXT_PUBLIC_SITE_URL,DB:!!process.env.DATABASE_URL,AUTH:!!process.env.AUTH_SECRET})"

```

Rezultat așteptat:

```

NODE_ENV=production/local conform mediului
NEXT_PUBLIC_SITE_URL corect
DB=true
AUTH=true

```

8. Smoke test automat rute

Rute publice critice:

```
/
/despre-noi/
/premii/
/servicii/
/noutati/
/cariere/
/contact/
/gdpr/
/politica-cookies/
/politica-confidentialitate/
/termeni-si-conditii/
/autentificare/
/recuperare-parola/
/sitemap.xml
/robots.txt
```

Rezultat așteptat:

- HTTP 200 sau redirect controlat;
- fără 500;
- fără 503;
- conținut HTML valid;
- titlu prezent;
- body non-gol.

9. Smoke test automat admin/portal

Rute protejate:

```
/admin/
/portal/
```

Rezultat așteptat neautentificat:

- redirect către autentificare;
- sau status controlat de acces;
- niciodată expunere de date private.

Rezultat așteptat autentificat:

- admin vede dashboard;
- client vede portal;
- rolurile sunt respectate.

10. Test automat API

10.1. ANAF

Input:

```
{"identifiant": "123456"}
```

Rezultat acceptabil:

- firmă găsită;
- sau eroare controlată dacă ANAF nu răspunde;
- nu 500 necontrolat.

10.2. Chat

Input:

```
{"message": "Bună ziua", "name": "Test", "email": "test@example.ro"}
```

Rezultat:

- sesiune creată;
- mesaj salvat;
- răspuns FAQ dacă există potrivire.

10.3. Easter egg

Rezultat:

- dacă este activ și în limite, apare token;
- dacă nu, răspuns controlat.

11. Test automat responsive

Viewporturi automate:

```
360x640  
390x844  
414x896  
768x1024  
1366x768  
1920x1080
```

Ce se verifică:

- nu există scroll orizontal excesiv;
- headerul nu se suprapune;
- footerul nu se rupe;
- cardurile se reaşază;
- butoanele sunt vizibile;
- admin sidebar rămâne utilizabil;
- portalul rămâne lizibil.

Rezultat:

- screenshot per rută;
- raport PASS/FAIL;
- listă diferențe vizuale.

12. Test automat SEO/GDPR basic

Pentru fiecare pagină:

- html lang;
- title;
- meta description;
- canonical;

- viewport;
- Open Graph;
- hreflang;
- imagini cu alt;
- linkuri legale în footer;
- cookie consent;
- header-e securitate.

Rezultat:

PASS dacă toate cerințele minime sunt prezente.
 WARN dacă lipsesc elemente necritice.
 FAIL dacă lipsesc elemente critice.

13. Raport automat recomandat

Format raport:

```
{
  "date": "2026-07-02T10:00:00+03:00",
  "target": "https://www.fiscontexpert.ro",
  "environment": "production",
  "summary": {
    "passed": 42,
    "failed": 2,
    "warnings": 5
  },
  "checks": [
    {
      "name": "Homepage status",
      "url": "/",
      "expected": "200",
      "actual": "200",
      "status": "PASS"
    }
  ]
}
```

14. Rezultate automate - tabel

Test	Comandă / tool	Așteptat	Obținut	Status
Install	npm install	fără erori	-	PASS/FAIL
Prisma	npm run db:generate	client generat	-	PASS/FAIL
Build	npm run build	build complet	-	PASS/FAIL
DB query	script Prisma	count returnat	-	PASS/FAIL
Homepage	HTTP GET /	200	-	PASS/FAIL
Admin	GET /admin neauth	redirect login	-	PASS/FAIL
Portal	GET /portal neauth	redirect login	-	PASS/FAIL
SEO	meta tags	prezente	-	PASS/WARN/FAIL
Responsive	screenshots	fără overflow	-	PASS/WARN/FAIL

15. Reguli de debugging

- 1 Nu repara la întâmplare.
- 2 Reproduce înainte să modifice.
- 3 Notează mediul exact: local/live/browser/device.
- 4 Verifică întâi configurația și datele, apoi codul.
- 5 Fixează strict problema raportată.
- 6 Nu atinge zone funcționale fără motiv.
- 7 Retestează bugul și zonele apropiate.
- 8 Actualizează documentația dacă se schimbă comportamentul.
- 9 Fă backup înainte de schimbări DB sau deploy.
- 10 Dacă bugul e pe live, activează mentenanța când impactul este mare.

16. Debugging pentru 503 pe hosting

Schema:

```
503 live
↓
Verifică Resource Usage cPanel
↓
Verifică procese node/lsnode
↓
Verifică runtime.log
↓
Verifică env DATABASE_URL / AUTH_SECRET / SITE_URL
↓
Verifică .next și asseturi
↓
Curăță public/_next dacă are permisiuni greșite
↓
Rulează npm install + db:generate
↓
Restart aplicație cPanel
↓
Testează /
```

Comenzi utile:

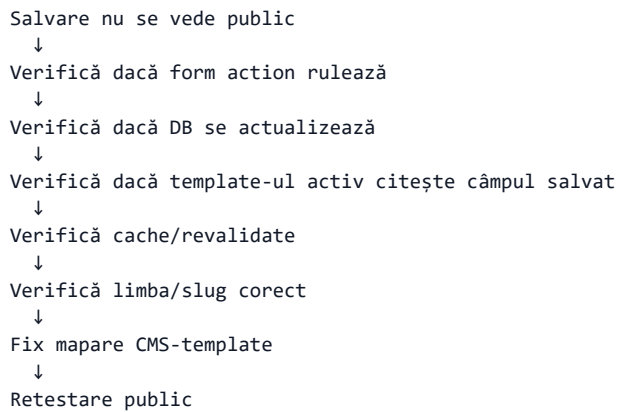
```
ps -u fiscont -f
kill -u fiscont -f lsnode
kill -u fiscont -f node
kill -u fiscont -f next
kill -u fiscont -f npm
tail -n 100 storage/runtime.log
rm -rf public/_next .next/dev
npm install
npm run db:generate
```

Rezultat așteptat:

- aplicația pornește;
- / răspunde;
- adminul are CSS;
- logul nu are erori critice.

17. Debugging salvare pagini

Schema:



Ce se testează:

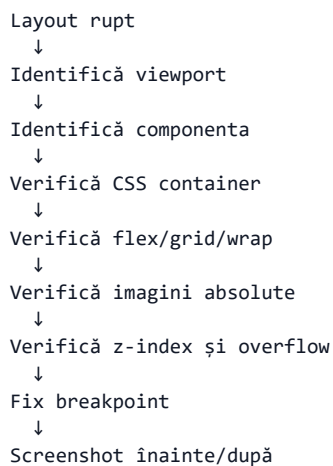
- text simplu;
- imagine;
- titlu SEO;
- meniu;
- secțiune JSON;
- template activ.

Rezultat așteptat:

Ce salvează adminul apare pe site în zona corespunzătoare.

18. Debugging responsive

Schema:



Rezultat așteptat:

- fără scroll orizontal;
- fără suprapuneri;
- meniul utilizabil;
- butoanele accesibile;
- text lizibil.

19. Debugging autentificare

Ce se testează:

- login admin;
- login client;
- parolă greșită;
- user inactiv;
- 2FA activ;
- cod 2FA greșit;
- cod expirat;
- resetare parolă;
- schimbare email;
- recuperare cont fără email.

Rezultate așteptate:

- mesaje clare;
- fără expunere dacă emailul există/nu există;
- sesiuni invalidate după reset;
- rolurile duc unde trebuie;
- user inactiv nu intră.

20. Debugging GDPR/ștergeri

Ce se testează:

- user fără date legale;
- user cu firmă;
- user cu documente;
- user cu payroll/taxe;
- firmă cu documente;
- firmă cu perioade contabile;
- backup snapshot existent;
- confirmare purge backup.

Rezultat așteptat:

- dacă există retenție legală, contul se dezactivează, nu se șterge total;
- dacă nu există retenție, datele se anonimizează/șterg conform procedurii;
- backup purge este marcat clar.

21. Debugging plăți

Ce se testează:

- modul dezactivat;
- modul activat;
- procesator selectat;
- preț perioadă;
- TVA;
- transfer bancar;
- card/metode online;
- creare comandă;
- confirmare.

Rezultat așteptat:

- dacă modulul e dezactivat, meniul Plătește nu apare;
- dacă e activ, formularul calculează totalul;
- transferul bancar creează comandă pending;
- metodele online fără gateway final nu trebuie prezentate ca plată confirmată automat.

22. Debugging chat

Ce se testează:

- widget activ/inactiv;
- poziție;
- culoare;
- formular contact;
- întrebare FAQ;
- fallback AI;
- răspuns admin;
- închidere conversație;
- link WhatsApp.

Rezultat așteptat:

- mesajul se salvează;
- FAQ răspunde când găsește potrivire;
- adminul vede conversația;
- WhatsApp este doar link dacă API-ul real nu este implementat.

23. Debugging integrări API

Ce se testează:

- salvare credentiale;
- criptare/stocare;
- status activ/inactiv;

- validare câmpuri;
- includere în analiză ca status.

Rezultat așteptat:

- credențialele nu se afișează în clar;
- setările persistă;
- sync real nu este presupus funcțional până nu este implementat providerul.

24. Concluzie

Debuggingul trebuie tratat ca proces, nu ca încercare rapidă.

Varianta manuală este obligatorie pentru:

- design;
- responsive;
- UX;
- conținut;
- probleme raportate vizual.

Varianta automată este obligatorie pentru:

- build;
- DB;
- rute critice;
- API-uri;
- regresii;
- deploy.

Regula finală:

Un bug este închis doar după reproducere, fix, retestare și documentarea rezultatului.