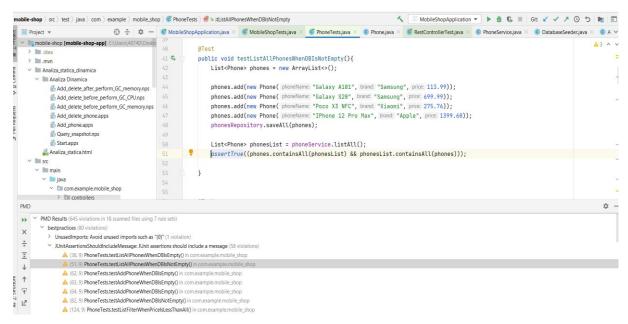
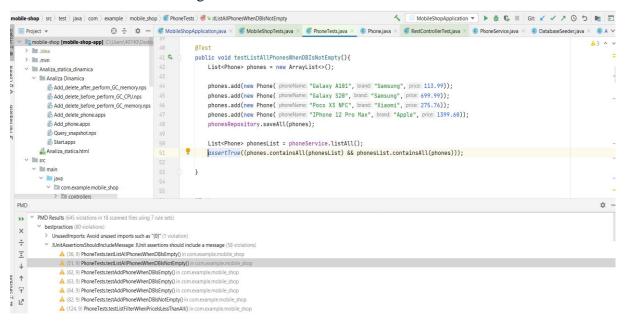
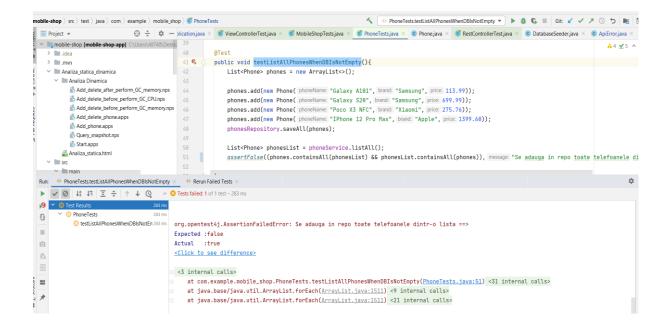
Analiza statica - cu PMD

 best practices – paranteze nefolositoare -> nu trebuie puse si la conditie, numai la assert

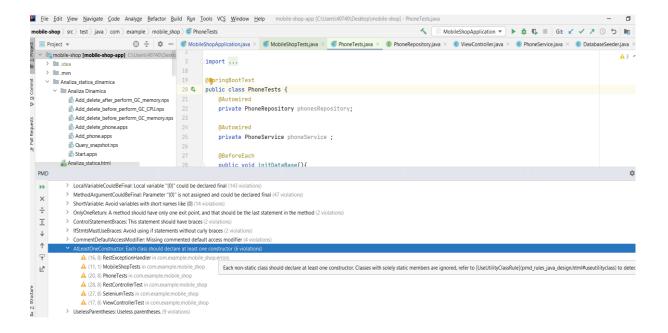


- best practices – assert-urile ar trebui sa contina un mesaj cu ce anume s-a testat pentru a identifica mai bine greseala in caz ca testul fail-uie





 codestyle – fiecare clasa ar trebui sa contina cel putin un constructor, dar clasele de test se presupune ca nu au nevoie sa fie instantiate, prin urmare nu consider acest lucru un bug



- design "incalcarea" legii lui Demeter, prin apelarea metodei matches pe un obiect al argumentului entity, si anume string-ul phoneName
- se rezolva prin adaugarea metodei emptyString care ia ca parametru phoneName si apeleaza pe el matches pentru a nu se face apelul pe phoneName returnat de un getter

```
bile-shop ⟩ src ⟩ main ⟩ java ⟩ com ⟩ example ⟩ mobile_shop ⟩ services ⟩ © PhoneService ⟩ m add
                                                                                                                                                                                                          ↑ PhoneTests.testListAllPhonesWhenDBIsNotEmpty ▼ ▶ # G | Git: ✓ ✓ ৴ ○ 5 | ■
                                                         🕃 😤 🗘 — Dication.java × 💰 ViewControllerTestjava × 💰 MobileShopTestsjava × 💰 PhoneErstsjava × 🚳 PhoneErstsjava × 🔞 PhoneErstsjava × 🔞 PhoneErstsjava × 🔞 PhoneErstsjava × 🔞 PhoneErstjava × 🔞 PhoneErstsjava × 🔞 PhoneErstsjava × 🐧
 mobile-shop [mobile-shop-app] C\Users\40740\Desktc 34
   > lim .idea
     ✓ ■ Analiza_statica_dinamica
                                                                                                          public Phone add(Phone entity) throws NegativeInputException, EmptyInputException {

∨ Ima Analiza Dinamica

                   Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica

Analiza Dinamica Dinamica

Analiza Dinamica Dina
                                                                                                                    if(entity.getPhoneName().matches( regex "^()**")){
    throw new EmptyInputException("phone input is empty!");
                   Add_delete_before_perform_GC_CPU.nps
                  Add_delete_before_perform_GC_memory.nps 40
                   Add_delete_phone.apps
                   Add_phone.apps
                                                                                                    if(entity.getBrand().matches( regex: "^( )*$")){
                   Ouery_snapshot.nps
                                                                                                                              throw new EmptyInputException("Brand input is empty!");
              Analiza_statica.html
                                                                                                            if(entity.getPrice() == null){
            ∨ 🖿 java
                                                                                                                              throw new NegativeInputException("Price input is empty!");
            > bestpractices (79 violations)
           > codestyle (271 violations)
÷
₹
                 > UseUtilityClass: All methods are static. Consider using a utility class instead. Alternatively, you could add a private constructor or make the class abstract to silence this warm... (1 violation)
                > DataClass: The class "(0)" is suspected to be a Data Class (WOC=(1), NOPA=(2), NOAM=(3), WMC=(4)) (2 violations) 
LawOfDemeter: Potential violation of Law of Demeter (120 violations)
                           A (38, 12) PhoneService.add() i
7
                         A (18, 9) MobileShopTests.contextLoads() in co
                           ▲ (36, 24) PhoneTests.testListAllPhonesWhenDBIsEmpty() in com.example.mobile_shop
                           ▲ (51, 55) PhoneTests.testListAllPhonesWhenDBIsNotEmpty() in com.example.mobile_shop
nobile-shop | src | main | java | com | example | mobile_shop | services | © PhoneService | @ PhoneService | @ PhoneTests.testListAllPhonesWhenDBisNotEmpty ▼ | ▶ 🎉 🕟 | | Git 🔏 🗸 🧷 🧐 | Illight 💉 🗸 🖔
                                                        🔞 😤 🌣 — xication.java × 💰 ViewControllerTestjava × 💰 MobileShopTestsjava × 💰 PhoneTestsjava × 🔞 PhoneService.java × 🔞 Phonejava × 💰 RestControllerTestjava × 💰 ApiError.java × 💃
 ■ Project ▼
 mobile-shop [mobile-shop-app] C\Users\40740\Deskto
    > idea
                                                                                                               private boolean emptyString(String stringToBeVerified) {
                                                                                                                      return stringToBeVerified.matches( regex: "^( )*$");
          ✓ Image Analiza Dinamica
                    Add_delete_after_perform_GC_memory.nps
                                                                                                              public Phone add(Phone entity) throws NegativeInputException, EmptyInputException {
                    Add delete before perform GC CPU.nps
                    Add_delete_before_perform_GC_memory.nps 42
                                                                                                                if(emptyString(entity.getPhoneName())) {
                                                                                                                                throw new EmptyInputException("phone input is empty!");
                    Add delete phone apps
                    Add_phone.apps
                    Cuery snapshot.nps
                    Start.apps
               Analiza statica.html
                                                                                                                        if(entity.getBrand().matches( regex "^( )*$")){
                                                                                                                                 throw new EmptyInputException("Brand input is empty!");
          ∨ 🗎 main

∨ I com.example.mobile_shop

PMD
 >> PMD Results (2 violations in 1 scanned file using 1 rule set)

    design (2 violations)

 <u>‡</u>
```

Analiza dinamica - cu Visual VM

 \downarrow

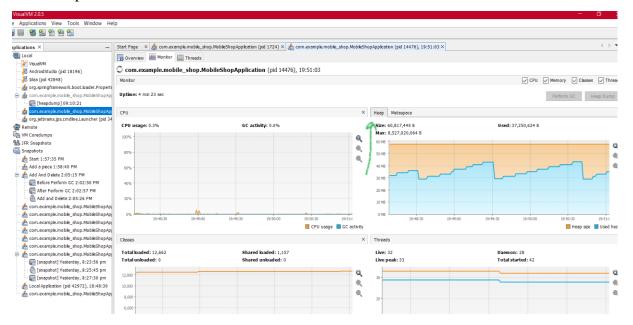
Ľ

In cazul de fata, nu ar exista probleme vizibile in ceea ce priveste consumul de memorie(HEAP, unde se stocheaza obiectele create) si timpul de raspuns al request-urilor.

Exemple de probleme:

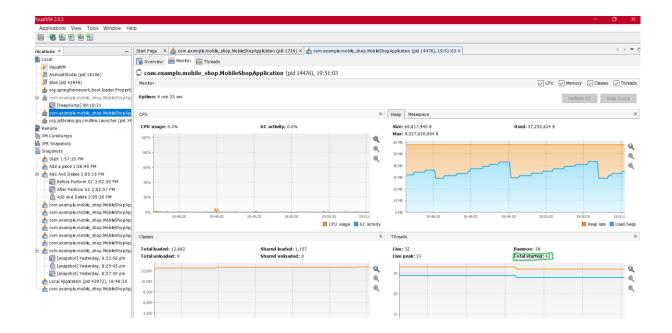
Un caz de incarcare a HEAP-ului ar fi creerea unor obiecte de dimensiuni mari, ceea ce nu este cazul pentru ca obiectele de tip telefon au ca si proprietati doar string-uri si un int, dar daca ar avea in componenta obiecte de mari dimensiuni ar ocupa multa memorie.

Deasemenea, foarte multe creari de obiecte noi ar incarca memoria. Utilizarea memoriei HEAP se poate observa aici:



O solutie ar fi configurarea dimensiunii HEAP-ului, prin setarea unor parametrii pentru JVM. De asemenea, se poate configura JVM-ul pentru optimizarea GC, in cazul in care se fac multe stergeri de exemplu. O cantitate mare de obiecte "de curatat", poate determina GC sa creasca latenta si sa foloseasca mai multe resurse. Se poate seta, de exemplu, ca GC sa ruleze concurent cu procesele aplicatiei si/sau sa "curete" cele mai recente obiecte din memorie.

Avand in vedere ca se foloseste serverul Tomcat, numarul de thread-uri care ruleaza in paralel este echivalent cu numarul de request-uri http facute catre server, in acelasi timp. Conectorul HTTP default se blocheaza, existand legatura 1 conexiune-1 thread-1 request. In cazul in care exista foarte multe thread-uri, se pot irosi resurse, pentru ca o conexiune nu e neaparat folosita la maximum. Cum nu am implementat posibilitatea crearii mai multor conturi pentru trimitere de request-uri de la mai multi useri si nici nu am generat mai multe request-uri concurente, cresterea semnificativa a numarului de thread-uri nu e vizibila in Visual VM la sectiunea Threads. Acest lucru s-ar observa aici:



O solutie ar fi configurarea serverului Tomcat pentru a folosi un conector NIO nonblocking, care pastreaza conexiunea activa pentru toate thread-urile create de request-uri. Dar, in ceea ce priveste timpul de executie, metoda blocking este mai rapida, dar daca exista posibilitatea mai multor conexiuni idle, e preferabila folosirea conectorului NIO.