

**ΤΕΙ ΠΕΙΡΑΙΑ**

**ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ**

**ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ**

**Εργασία Εξαμήνου για Μηχανική Λογισμικού**

**Σύστημα διαχείρισης αποθήκης καταστήματος με ηλεκτρονικά**

Ι. Δρυμούσης , Ζ. Δημητριάδης , Γ. Μαθιουδάκης

**ΑΙΓΑΛΕΩ**

**Ιανουάριος 2015**

**ΕΡΓΑΣΙΑ ΕΞΑΜΗΝΟΥ**

Σύστημα διαχείρισης αποθήκης καταστήματος με ηλεκτρονικά

**Δρυμούσης Ιωάννης**

**Α.Μ. 42016**

**Δημητριάδης Ζαχαρίας**

**Α.Μ. 42080**

**Μαθιουδάκης Γεώργιος**

**Α.Μ. 42041**

**Εισηγητής:**

**Ευάγγελος Κοσμάτος**

**Ημερομηνία παράδοσης 27/1/2015**

**ΠΕΡΙΛΗΨΗ**

Η παρούσα εργασία έχει ως σκοπό την ανάπτυξη ενός συστήματος διαχείρισης αποθήκης καταστήματος με ηλεκτρονικά . Στην παρουσίαση που ακολουθεί , θα δούμε το πώς γίνεται ολοκληρωμένα , από την αρχή μέχρι το τέλος , η ανάπτυξη ενός τέτοιου λογισμικού περνώντας από τα διάφορα στάδια ανάπτυξης ( Ανάλυση απαιτήσεων , UML διαγράμματα , Debugging , κτλ. ).

Πίνακας περιεχομένων

**Εισαγωγή4**

**Ανάλυση Απαιτήσεων**5

**Διαγράμματα UML5**

Διαγράμματα Περιπτώσεων5

Διαγράμματα Δραστηριοτήτων6

Διαγράμματα Ακολουθίας9

Διαγράμματα Κλάσεων12

**Πηγαίος κώδικας*14***

Screenshots από GUI14

Screenshots από debugging17

**Αποτελέσματα static analysis19**

**Unit Testing20**

Code Coverage20

**User Acceptance Test21**

**Υποσημειώσεις21**

**ΕΙΣΑΓΩΓΗ**

Στην επιστήμη της πληροφορικής προκειμένου να αναπτυχθεί μια εφαρμογή χρειάζονται κάποια στάνταρτ βήματα μέχρις ότου αυτή ολοκληρωθεί. Τα βήματα αυτά ή καλύτερα φάσεις , που περνάει ένα πρόγραμμα προκειμένου να φτάσει από την σύλληψη της ιδέας στην παραγωγή είναι αυστηρά ορισμένα , αποδεκτά και υλοποιούνται από επαγγελματίες προγραμματιστές και εταιρίες πληροφορικής σε ολόκληρο τον κόσμο.

Σκοπός αυτών των φάσεων είναι να οργανώσουν την δουλειά που απαιτείται για την δημιουργία μιας εφαρμογής , με γνώμονα την ελαχιστοποίηση των αποκλίσεων από την αρχική ιδέα και την διασφάλιση της ποιότητας στο τελικό προϊόν. Ένας ακόμα λόγος τήρησης αυτών των φάσεων είναι η αποσαφήνιση των απαιτήσεων και η εγκαθίδρυση ενός κοινού τόπου επικοινωνίας με τον πελάτη. Τέλος μπορεί να θεωρηθεί και ως ένα άτυπο συμβόλαιο μεταξύ του προγραμματιστή και του πελάτη για το τι έχει συμφωνηθεί και το πώς αυτό θα υλοποιηθεί.

Οι φάσεις αυτές λοιπόν , που ολοκληρωμένα συνθέτουν το πεδίο της Μηχανικής Λογισμικού είναι οι ακόλουθες. :

* Ανάλυση απαιτήσεων
* Διαγράμματα UML
* Ανάπτυξη κώδικα
* Debugging
* Static Analysis
* Unit Testing
* Code Coverage
* Acceptance Testing

Στην συνέχεια της παρουσίασης θα δούμε σαν παράδειγμα, όλες αυτές τις φάσεις υλοποιημένες για το δικό μας project.

**ΑΝΑΛΥΣΗ ΑΠΑΙΤΗΣΕΩΝ**

Στην φάση της ανάλυσης απαιτήσεων καταγράφουμε τις λειτουργικές και μη λειτουργικές απαιτήσεις της εφαρμογής μας. Ορίζουμε επίσης αναλυτικά τις διάφορες λειτουργίες του προγράμματός μας , τους χρήστες και τα δικαιώματα που έχουν, τις διεπαφές χρήστη, υλικού , λογισμικού , επικοινωνιών και το περιβάλλον λειτουργίας με πιθανούς περιορισμούς και εξαρτήσεις. Τέλος η ανάλυση απαιτήσεων αποτελεί τον κοινό τρόπο επικοινωνίας και το συμβόλαιο μεταξύ του προγραμματιστή και του πελάτη καθώς εκεί αποσαφηνίζονται όλες οι λεπτομέρειες της εφαρμογής.

Η δικιά μας ανάλυση απαιτήσεων βρίσκεται σε ένα **ξεχωριστό word** αρχείο στον φάκελο που στείλαμε την εργασία, με τίτλο ***Ανάλυση απαιτήσεων*** *.*Ο λόγος για τον οποίο δεν συμπεριλήφθη στο παρόν έγγραφο είναι , ότι επιθυμούσαμε μια πιο λεπτομερή και καθαρή προβολή της συγκεκριμένης πολύ βασικής φάσης.

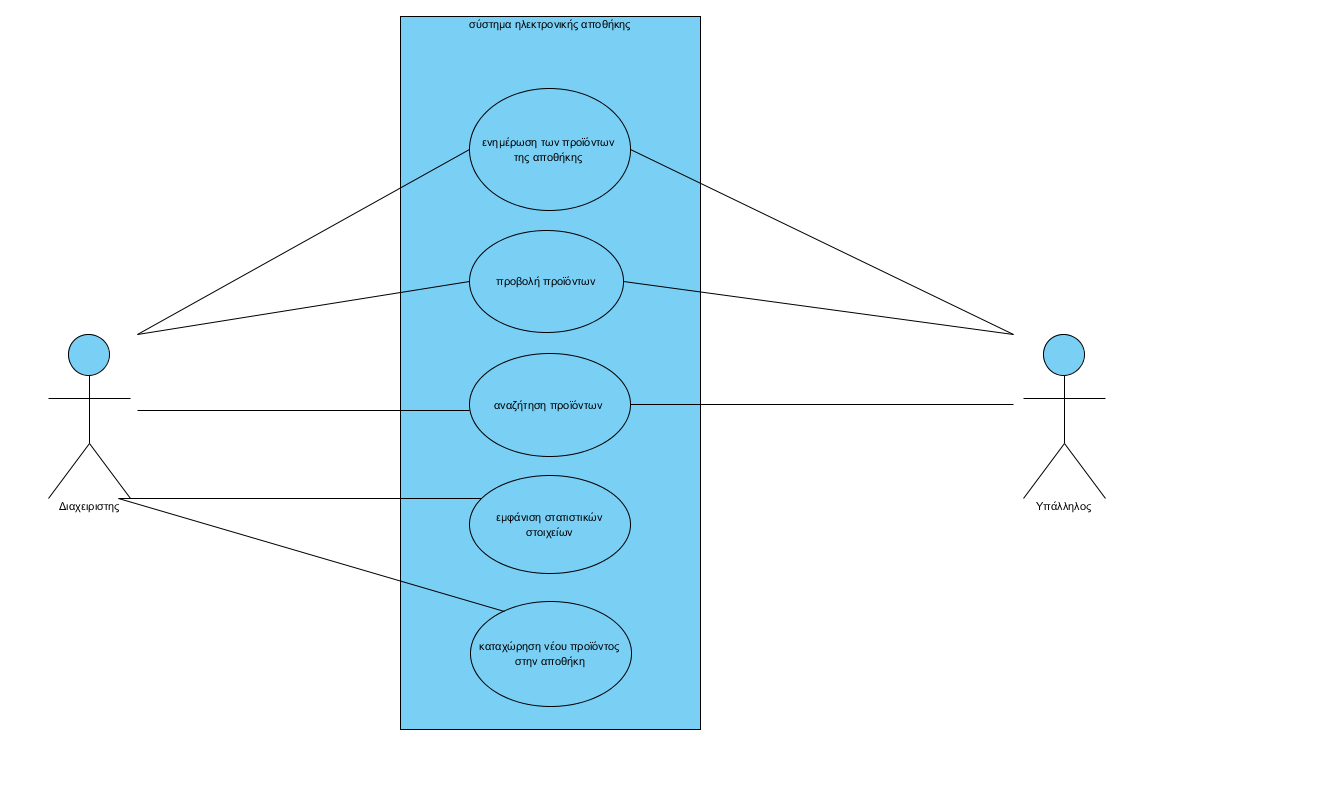
**ΔΙΑΓΡΑΜΜΑΤΑ UML**

Η Unified Modeling Language (UML) είναι η πρότυπη γλώσσα μοντελοποίησης στη μηχανική λογισμικού. Χρησιμοποιείται για τη γραφική απεικόνιση, προσδιορισμό, κατασκευή και τεκμηρίωση των στοιχείων ενός συστήματος λογισμικού. Μπορεί να χρησιμοποιηθεί σε διάφορες φάσεις ανάπτυξης, από την ανάλυση απαιτήσεων ως τον έλεγχο ενός ολοκληρωμένου συστήματος. Αποτελείται από ένα σύνολο προσυμφωνημένων όρων, συμβόλων και διαγραμμάτων όπως τα διαγράμματα περιπτώσεων, ακολουθίας, δραστηριοτήτων , κλάσεων και άλλα. Για την ανάλυση ενός συστήματος διαχείρισης αποθήκης με ηλεκτρονικά χρησιμοποιήσαμε τα διαγράμματα περιπτώσεων, ακολουθίας ,δραστηριοτήτων και κλάσεων.

**Διάγραμμα Περιπτώσεων ( Case Diagram )**

Τα διαγράμματα περιπτώσεων χρήσης παρουσιάζουν τους actors και τις περιπτώσεις χρήσης ενός συστήματος μαζί με τις σχέσεις μεταξύ τους.

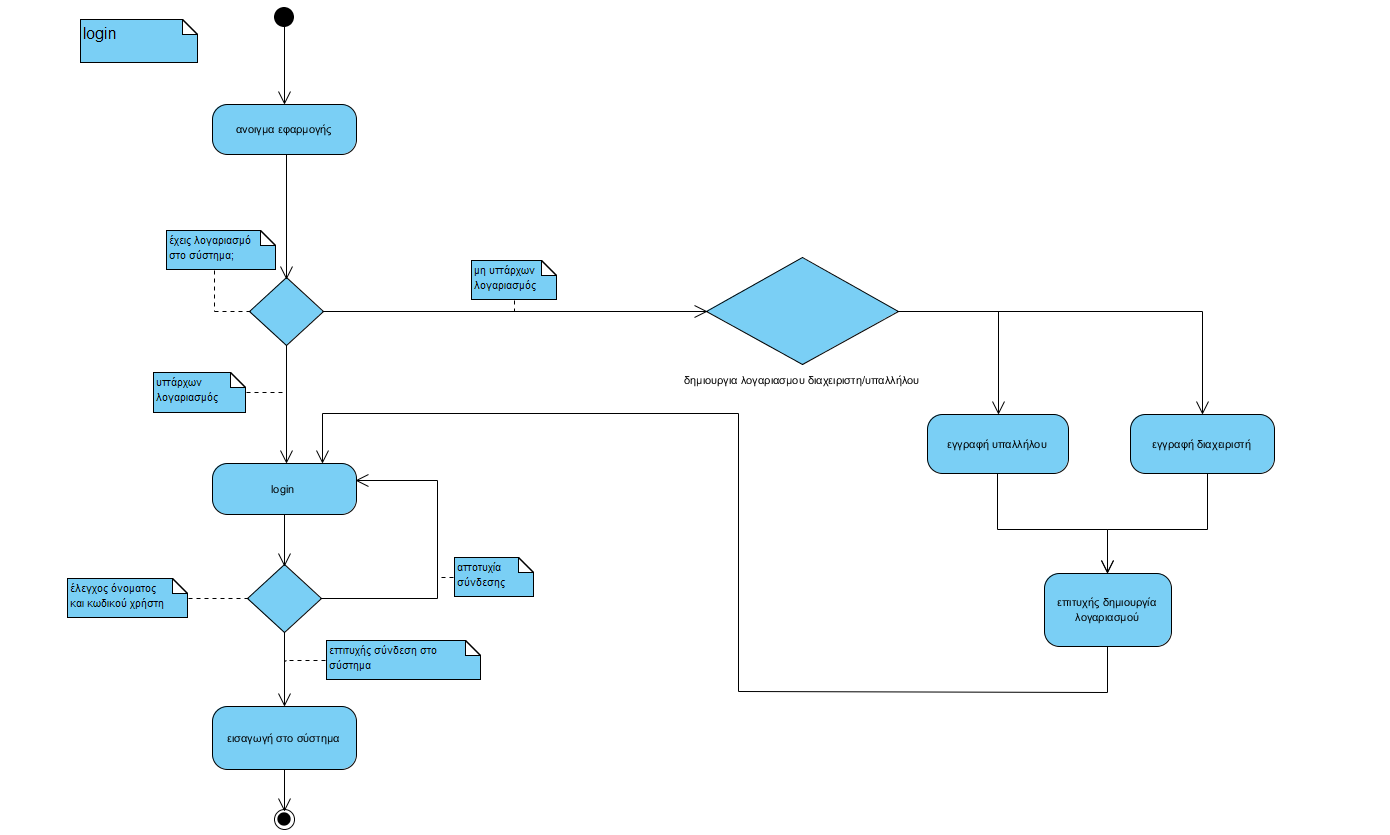
*Διάγραμμα περιπτώσεων χρήσης του Project μας*



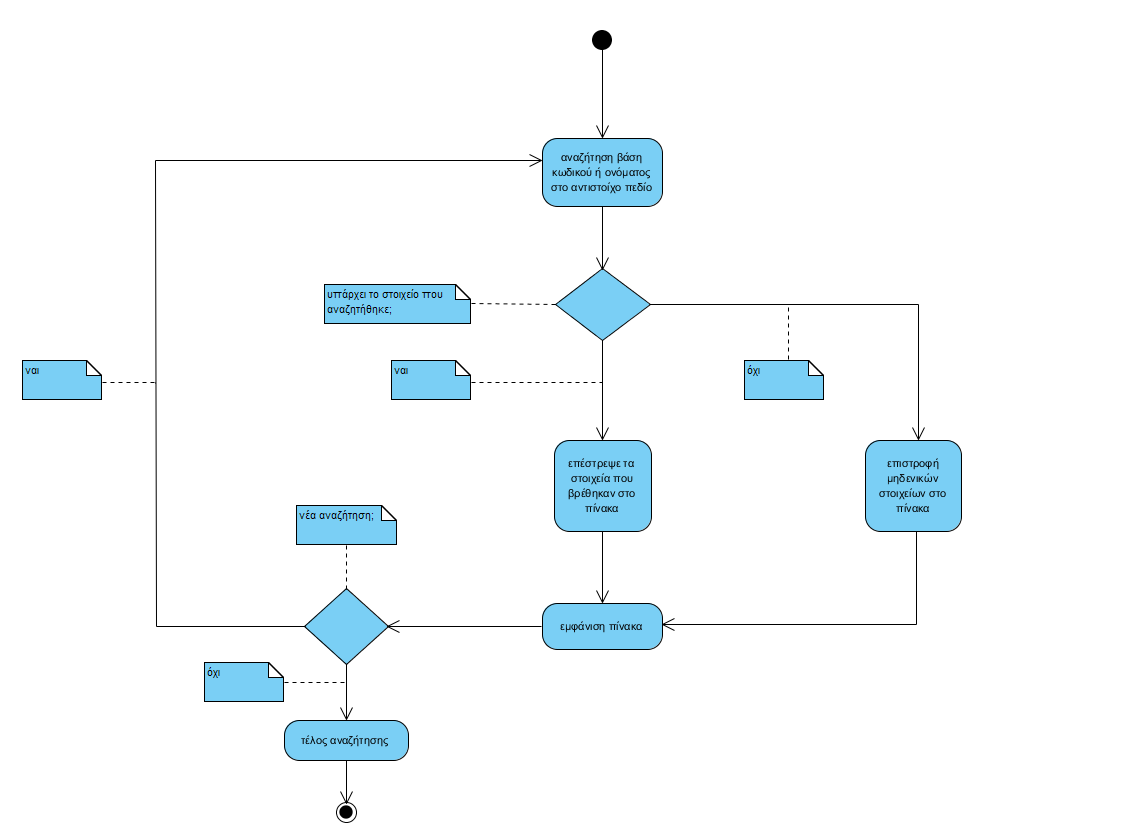
**Διαγράμματα Δραστηριοτήτων ( Activity Diagrams )**

Τα Διαγράμματα δραστηριότητας (Activity diagrams) είναι γραφικές αναπαραστάσεις των ροών εργασίας (workflows) των σταδιακών δραστηριοτήτων και δράσεων με υποστήριξη για επιλογή, επανάληψη και συγχρονισμό. Στην UML (Unified Modeling Language), είναι διαγράμματα που προορίζονται να μοντελοποιήσουν τόσο τις υπολογιστικές όσο και τις οργανωτικές διαδικασίες (π.χ. ροές εργασίας). Για την υλοποίηση του έργου μας χρησιμοποιήσαμε 5 διαγράμματα για να περιγράψουμε τις οργανωτικές διεργασίες.

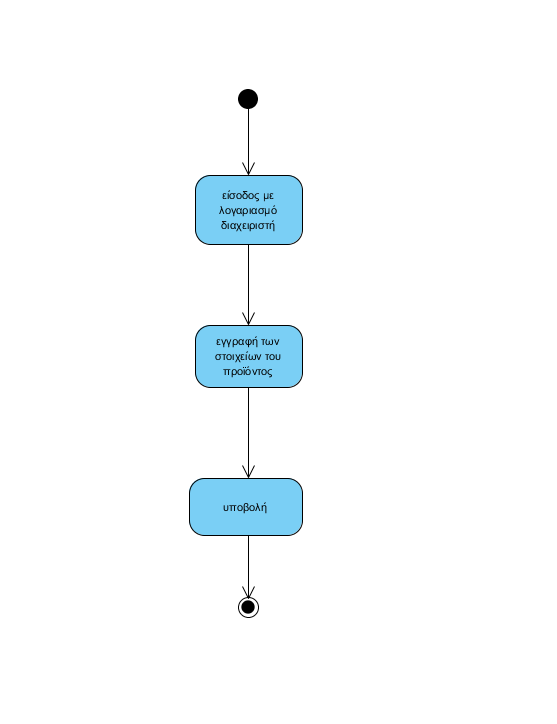
*Διάγραμμα περιγραφής εισόδου χρήστη( διαχειριστής, υπάλληλος ) στο σύστημα*



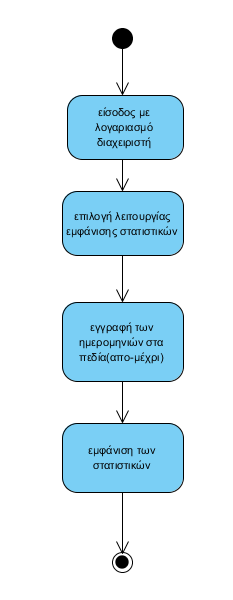
*Διάγραμμα του τρόπου λειτουργίας της αναζήτησης*



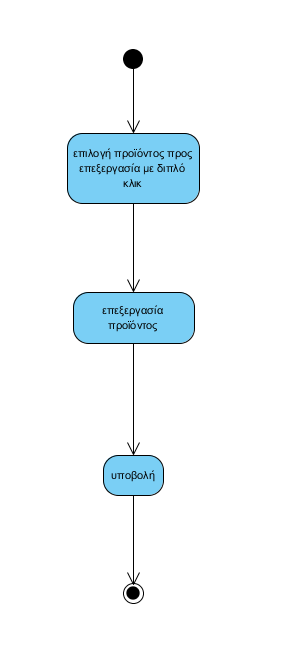
*Διάγραμμα του τρόπου εισαγωγής νέου προϊόντος στην αποθήκη*



*Διάγραμμα της λειτουργίας εμφάνισης στατιστικών*



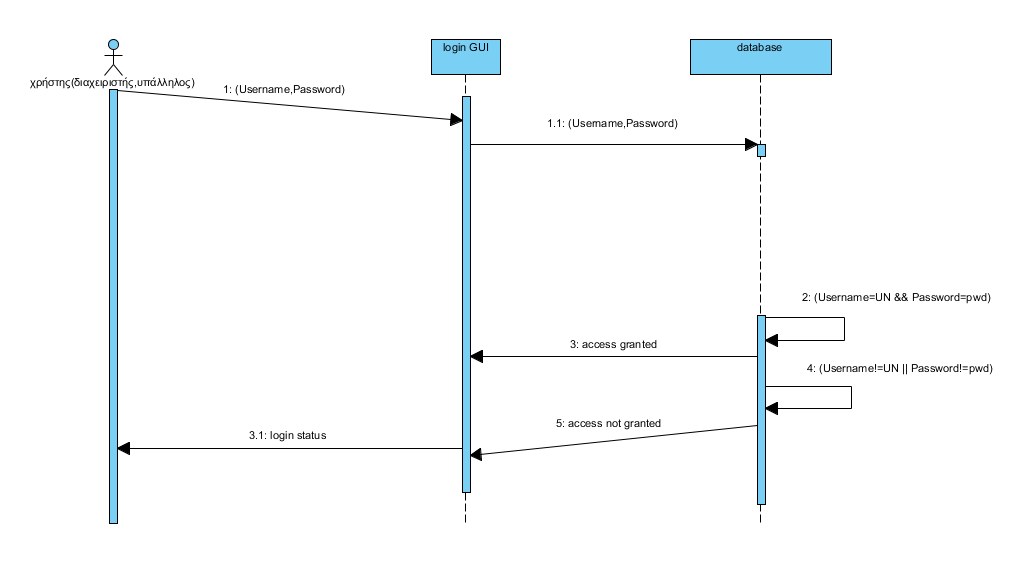
*Διάγραμμα του τρόπου επεξεργασίας της αποθήκης*



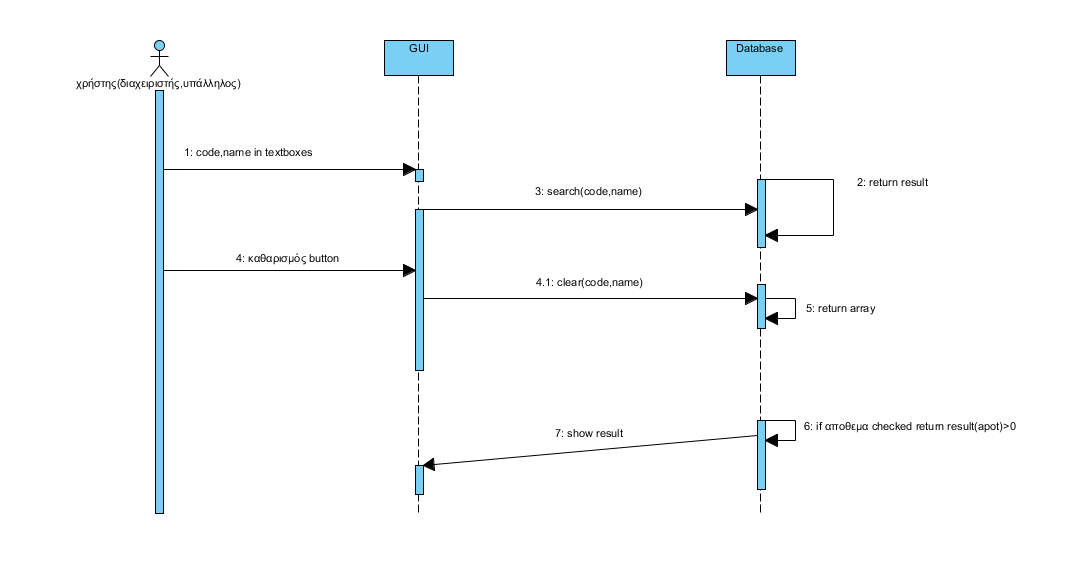
**Διαγράμματα Ακολουθίας ( Sequence Diagrams )**

Τα διαγράμματα ακολουθίας παρουσιάζουν την αλληλεπίδραση των αντικειμένων µέσω της ανταλλαγής μηνυμάτων.

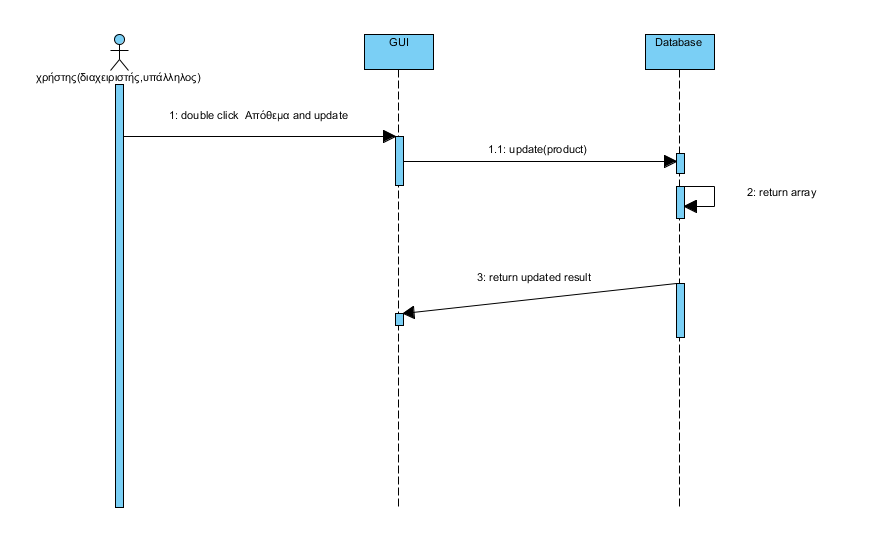
*Διάγραμμα ακολουθίας για την είσοδο ενός χρήστη*



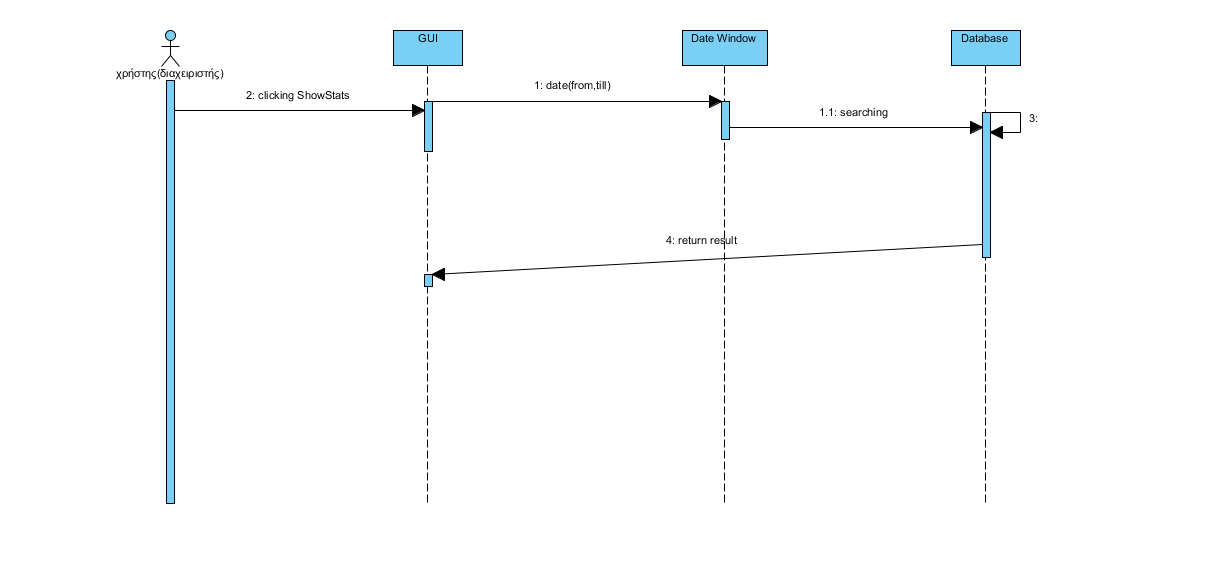
*Διάγραμμα ακολουθίας αναζήτησης ενός προϊόντος με βάση το όνομα ή το κωδικό*



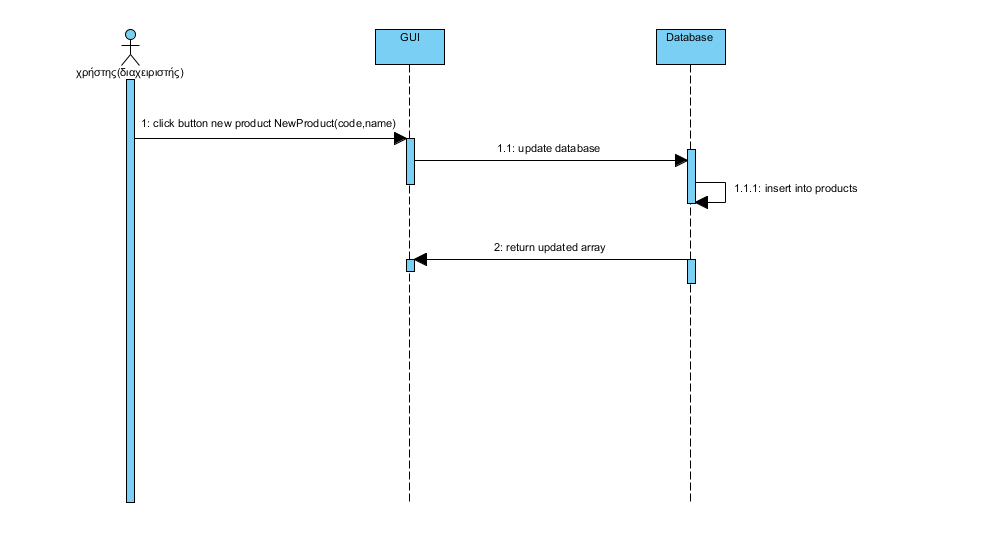
*Διάγραμμα ακολουθίας επεξεργασίας αποθέματος των προϊόντων της αποθήκης*



*Διάγραμμα ακολουθίας εμφάνισης στατιστικών στοιχείων μόνο για το διαχειριστή*



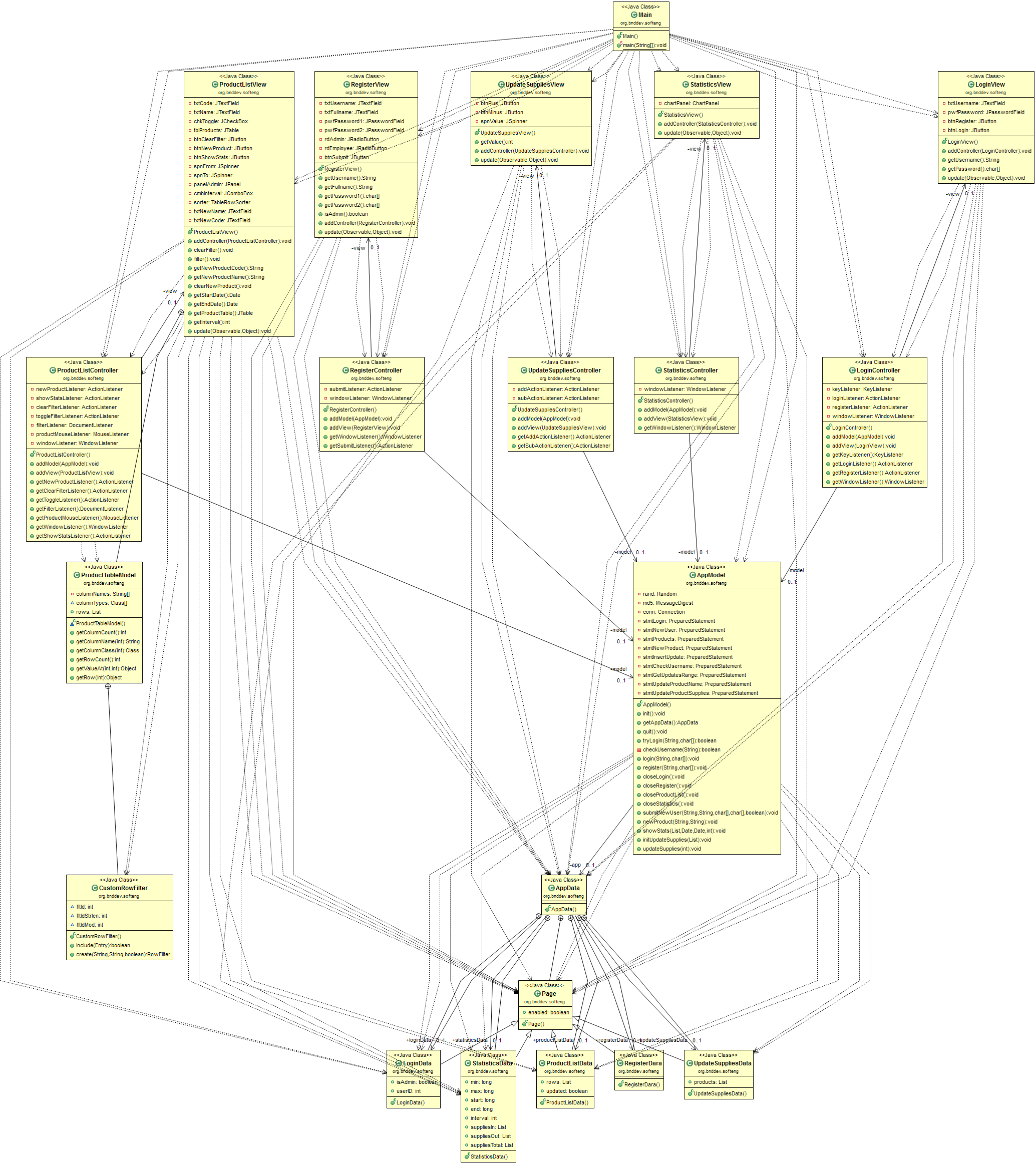
*Διάγραμμα ακολουθίας εισαγωγής νέου προϊόντος μόνο από το διαχειριστή*



**Διάγραμμα κλάσεων ( Class Diagram )**

Τα διαγράμματα κλάσεων της UML ορίζουν γεωμετρικά σχήματα ως συμβολισμούς για τα αντικείμενα, τις κλάσεις και τις διασυνδέσεις ενώ διαφόρων τύπων γραμμές χρησιμοποιούνται για να συνδέουν αυτά τα σχήματα και να υποδηλώνουν έτσι τον τρόπο που κληρονομούν, συνεργάζονται ή εξαρτώνται μεταξύ τους. Τα αντικείμενα της ίδιας κλάσης αναπαρίστανται με ένα μόνο γεωμετρικό σχήμα. Όταν ένα αντικείμενο χρησιμοποιεί κώδικα κάποιας άλλης κλάσης (π.χ. καλώντας μία μέθοδό της), σύμφωνα με το πρότυπο της UML υπάρχει μία «εξάρτηση» ( dependency ) μεταξύ τους η οποία αναπαρίσταται με μία διακεκομμένη γραμμή.

*Διάγραμμα κλάσης*

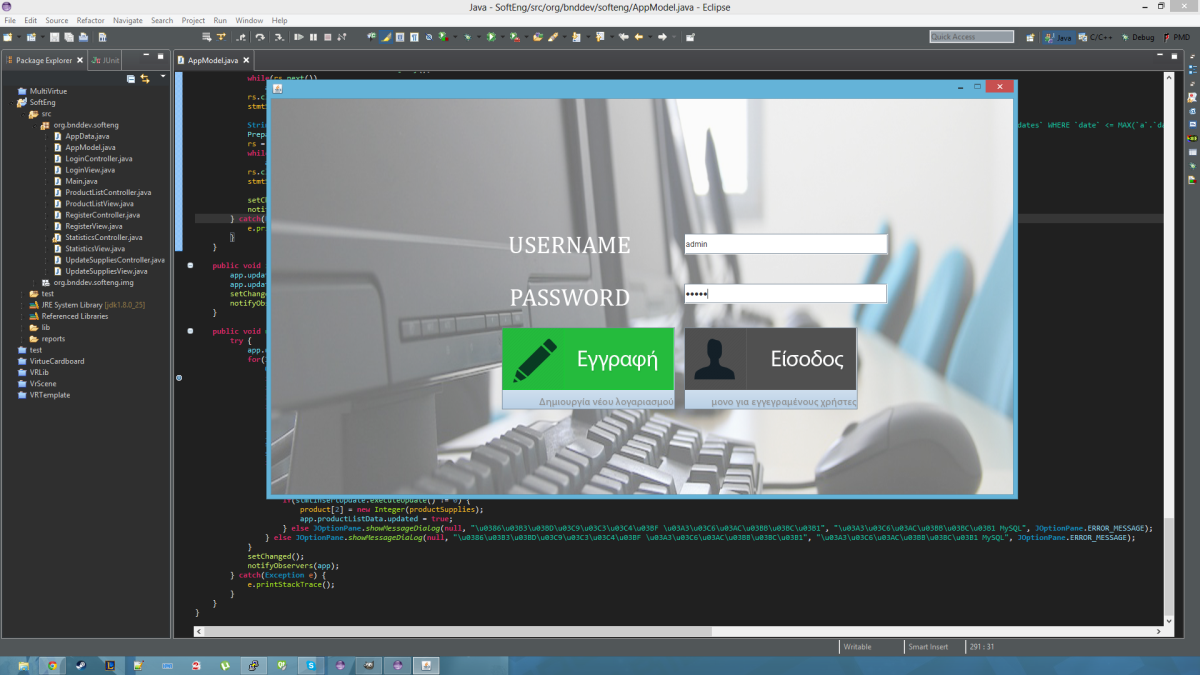


**ΠΗΓΑΙΟΣ ΚΩΔΙΚΑΣ**

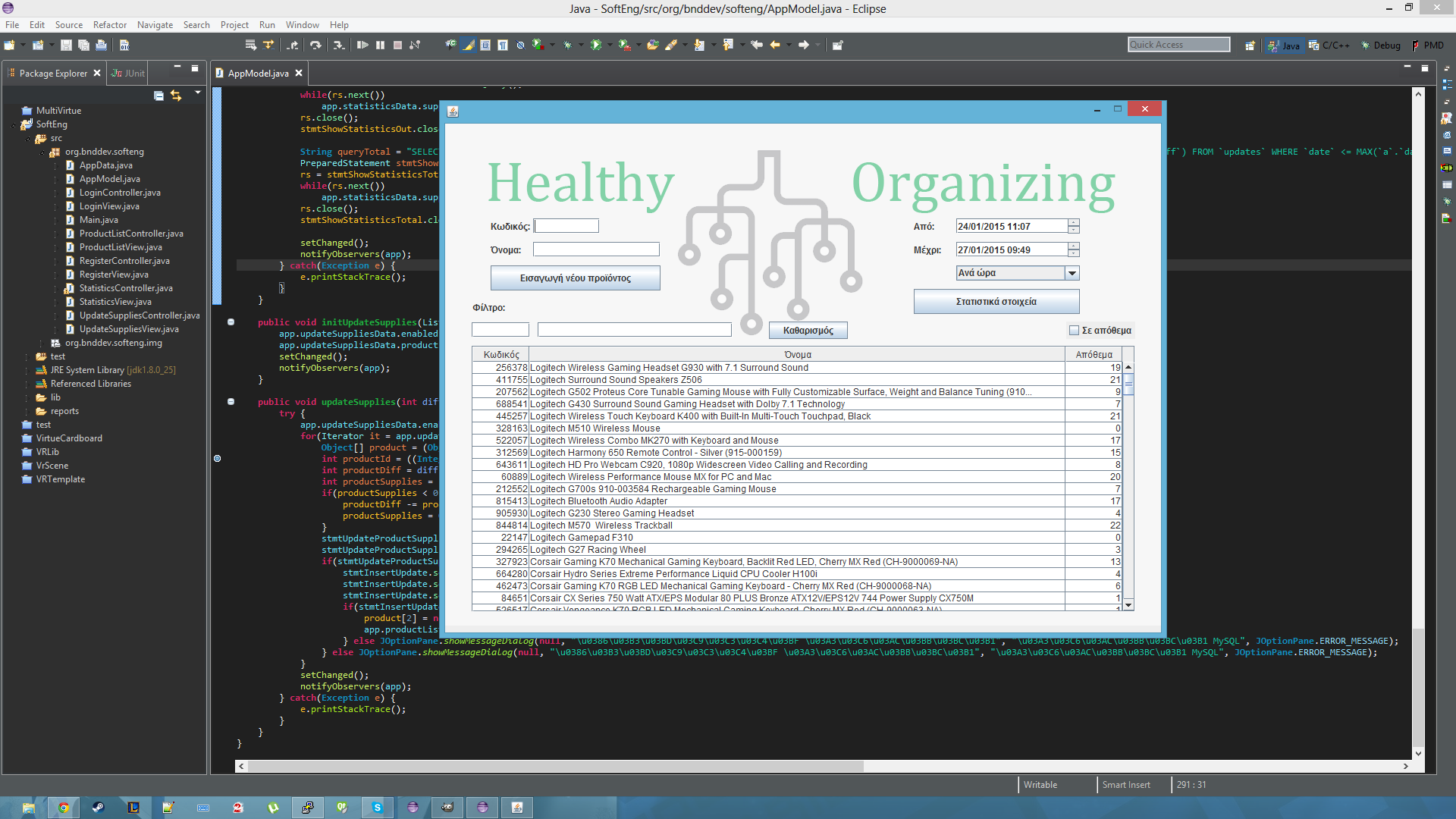
Σε αυτή την φάση είμαστε έτοιμοι να αναπτύξουμε τον κώδικα της εφαρμογής μας. Για λόγους οργανωτικούς και χωροταξικούς ο κώδικας δεν συμπεριλαμβάνεται στο παρόν έγγραφο, αλλά δίνεται σαν εξωτερικός φάκελος μέσα στον φάκελο της εργασίας που στείλαμε, με το όνομα ***SoftEng*** . Στον φάκελο αυτό επίσης συμπεριλαμβάνεται και το αρχείο ***softeng.mysql*** που αποτελεί το backup της βάσης μας.

**Screenshots από το GUI**

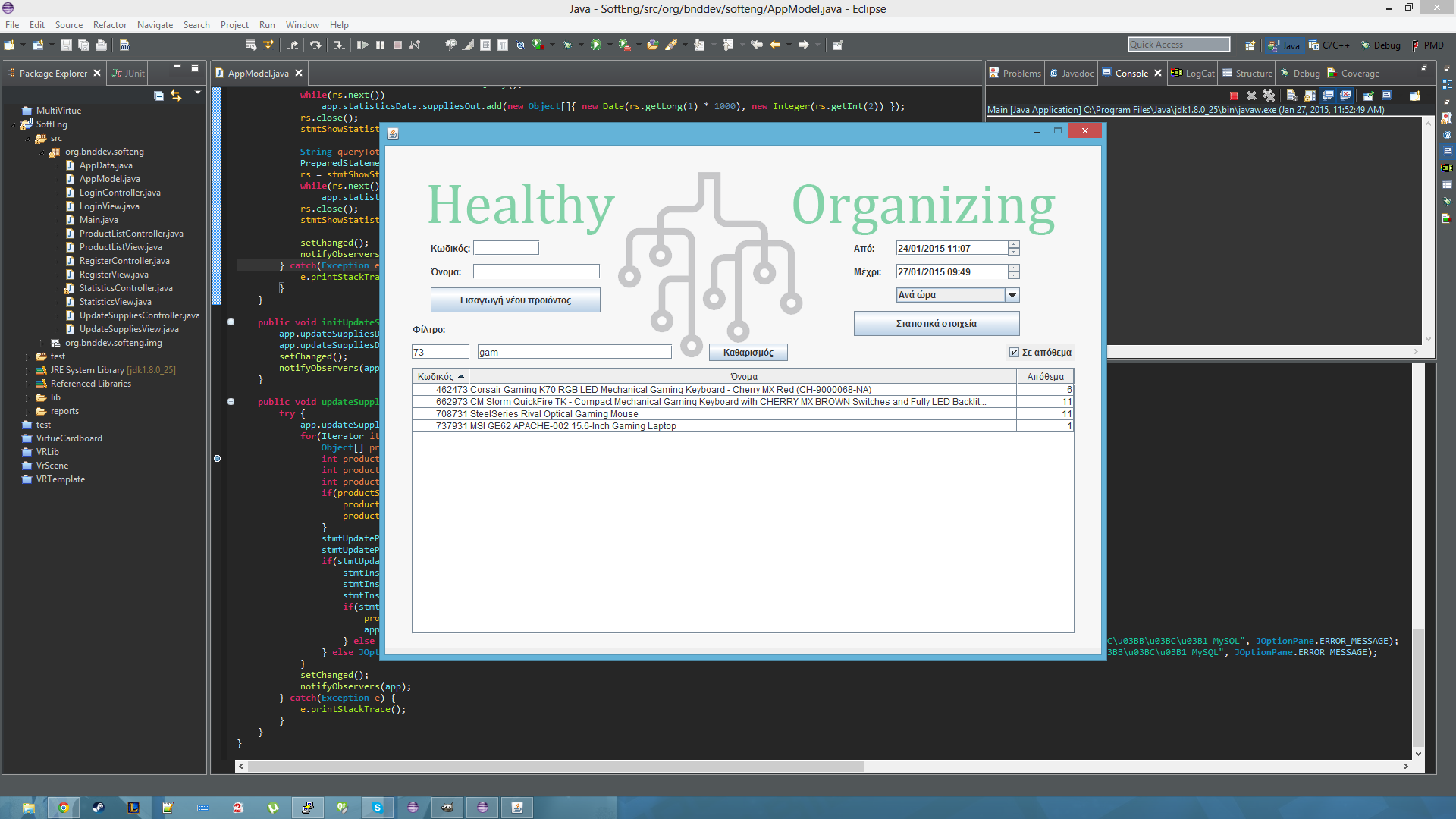
GUI login



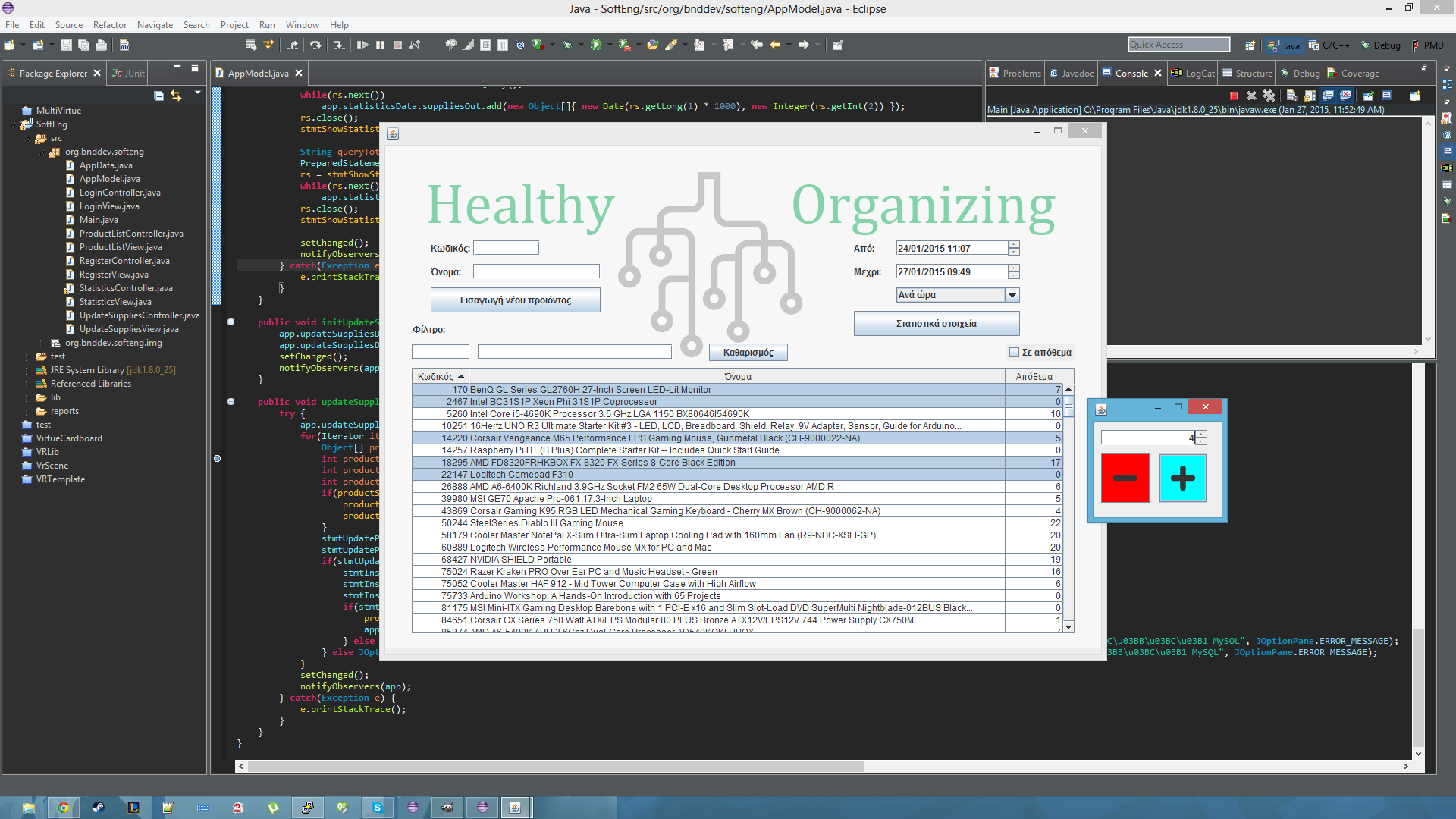
Βασικό GUI διαχειριστή



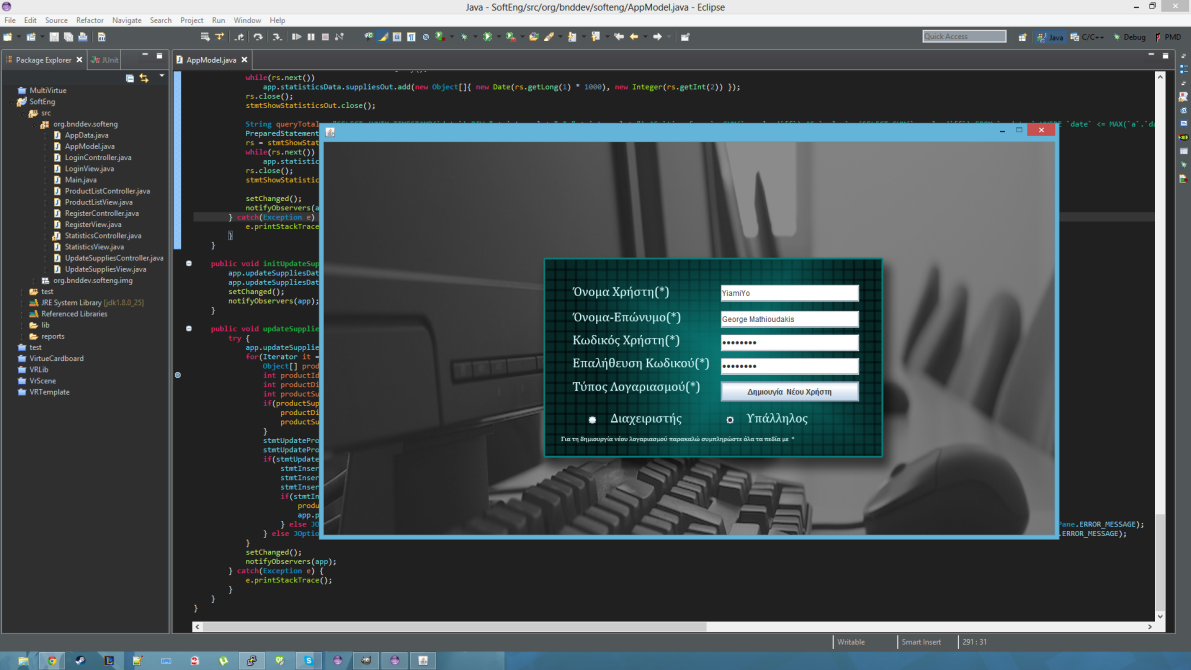
Στιγμιότυπο GUI κατά την αναζήτηση προϊόντος



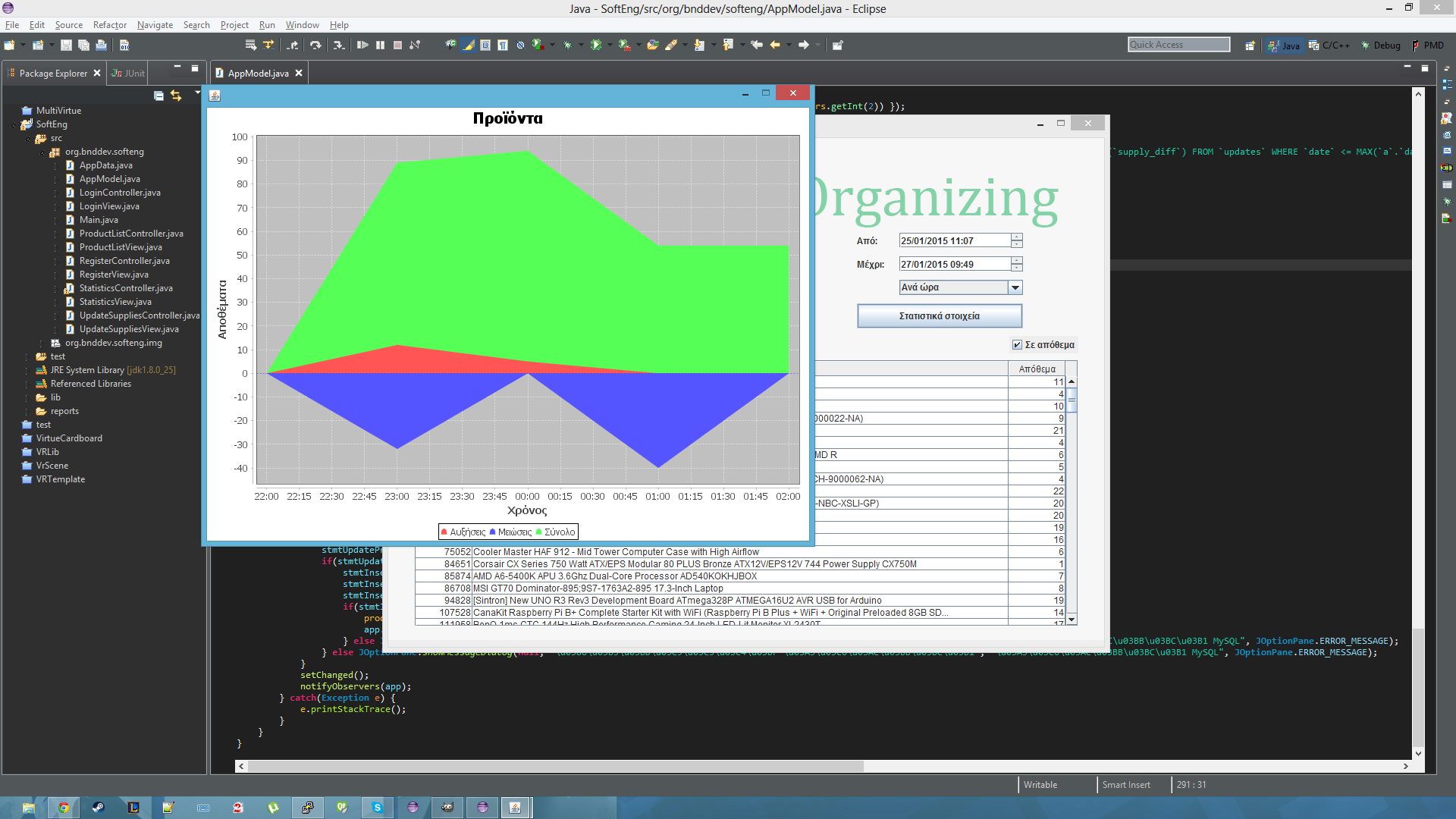
Στιγμιότυπο GUI κατά την επεξεργασία προϊόντος



GUI register νέου χρήστη

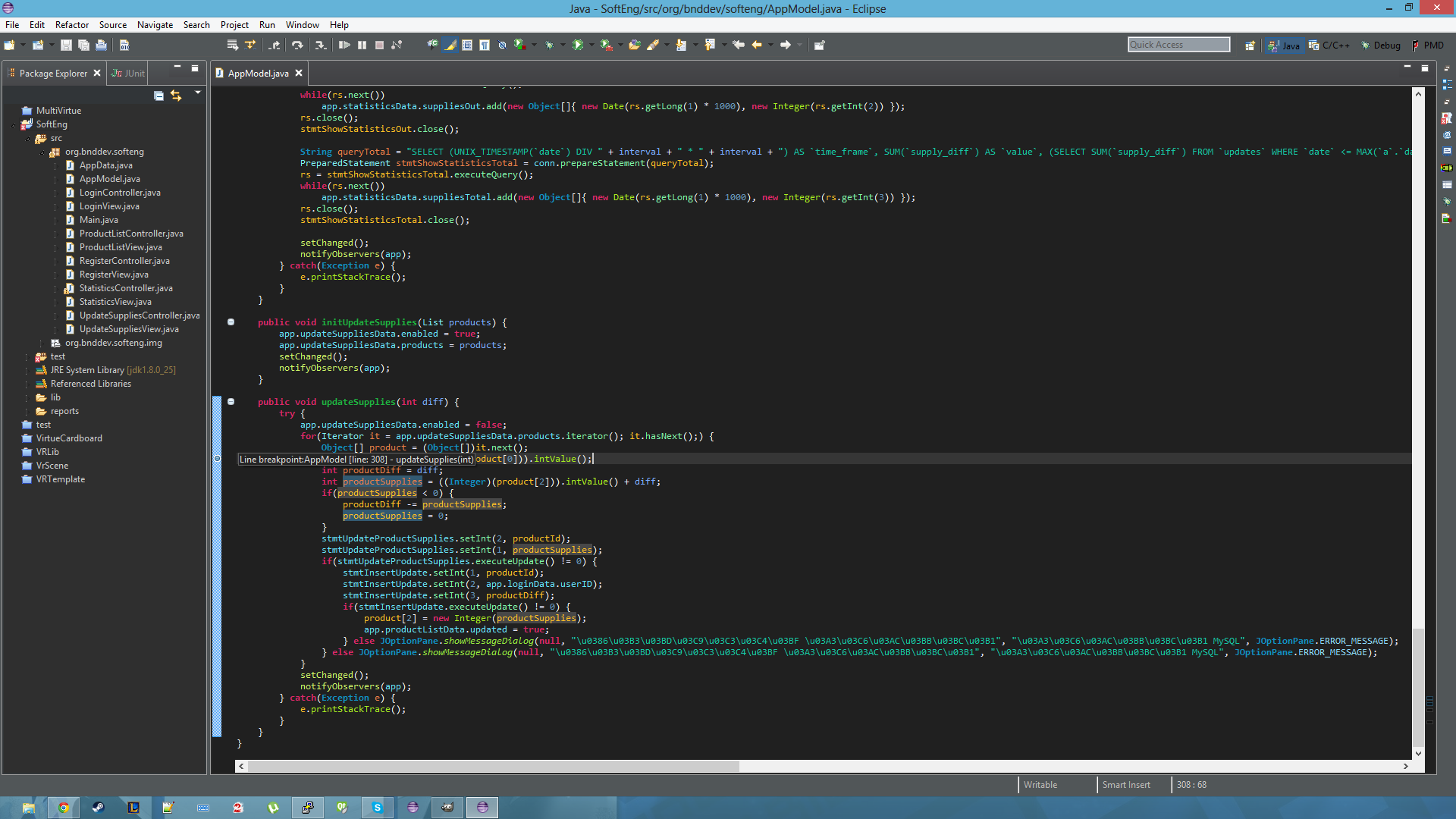


GUI στατιστικών διαγραμμάτων

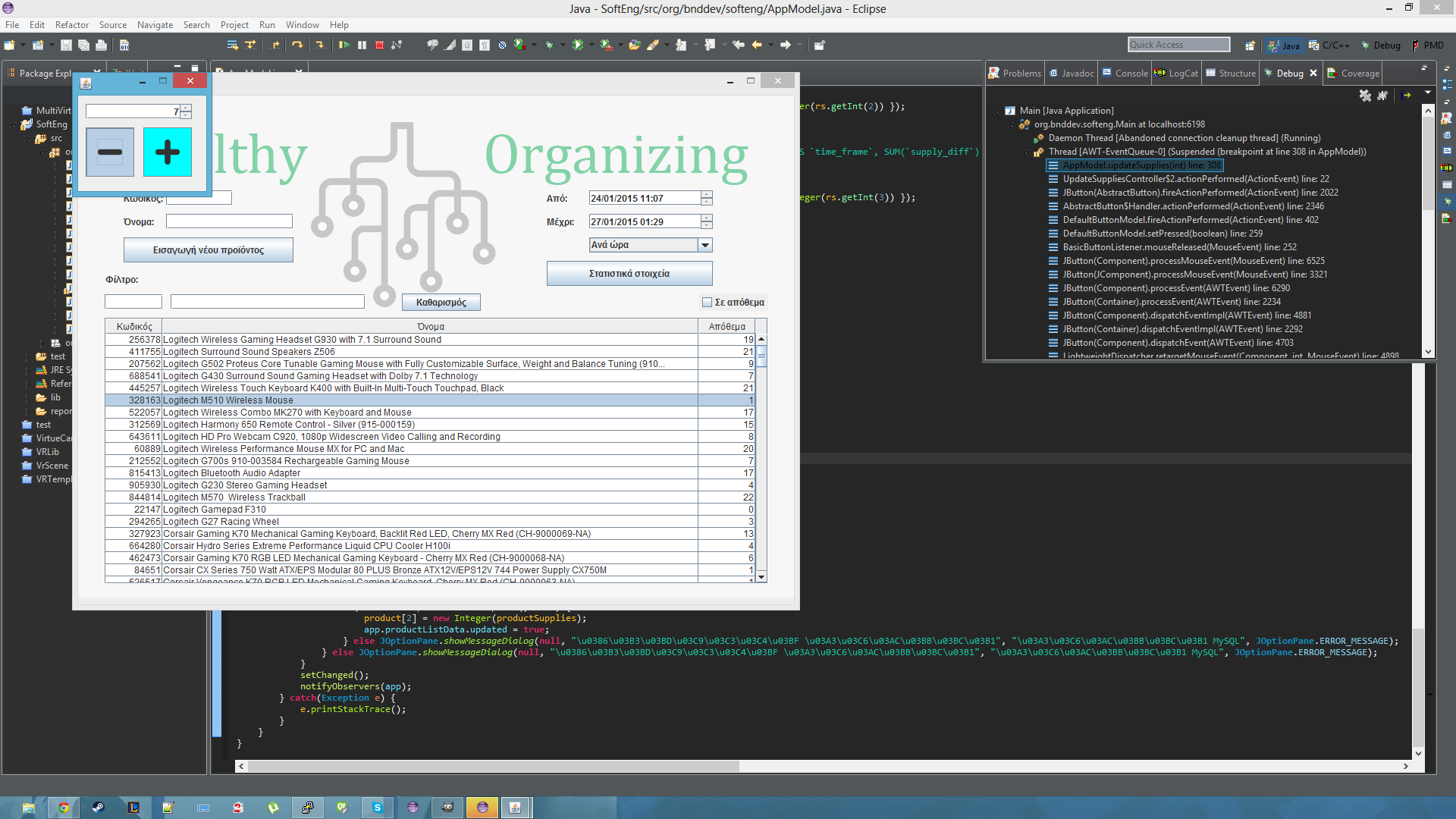


**Screenshots από το debugging**

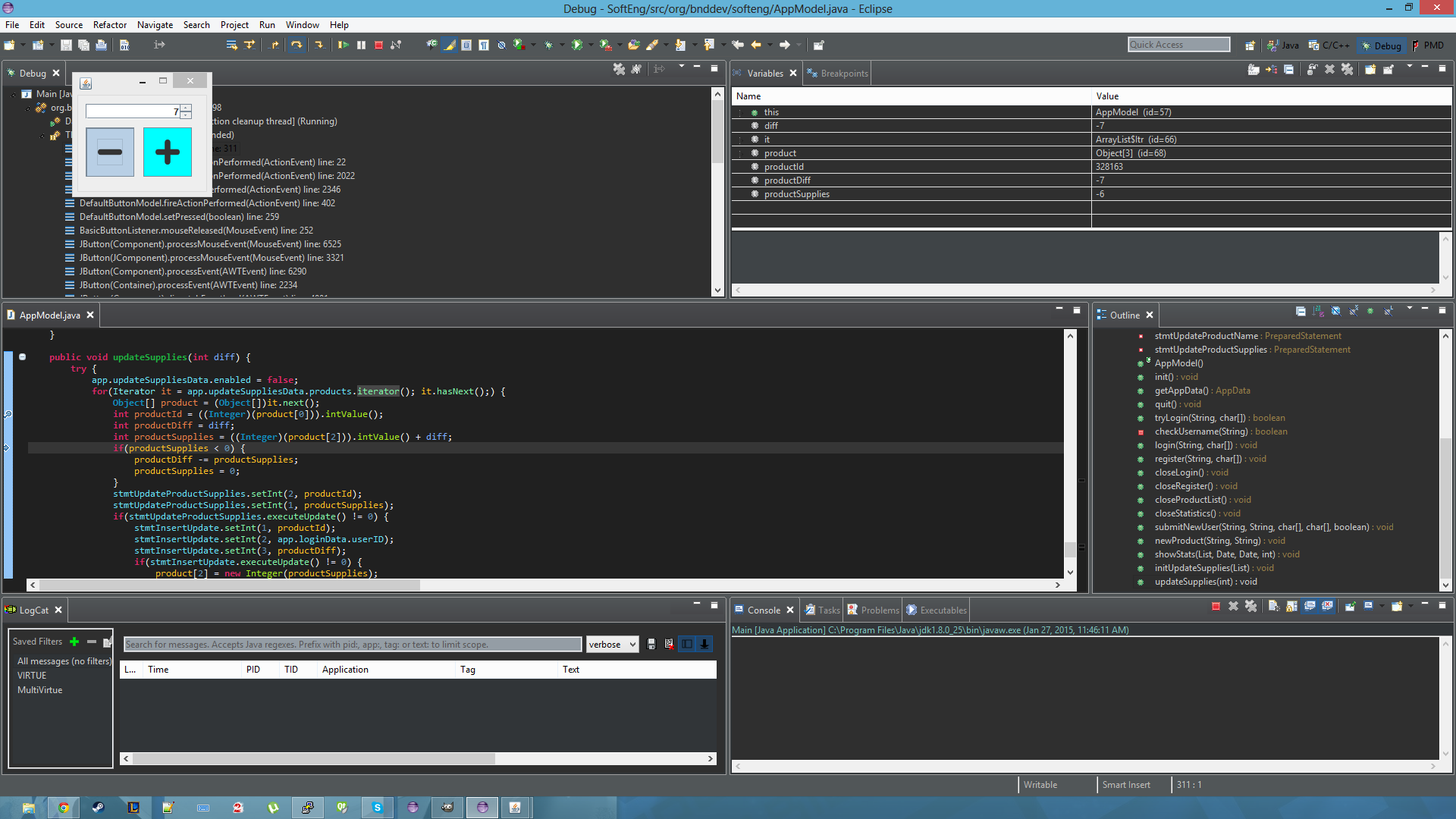
Εισαγωγή Breakpoint



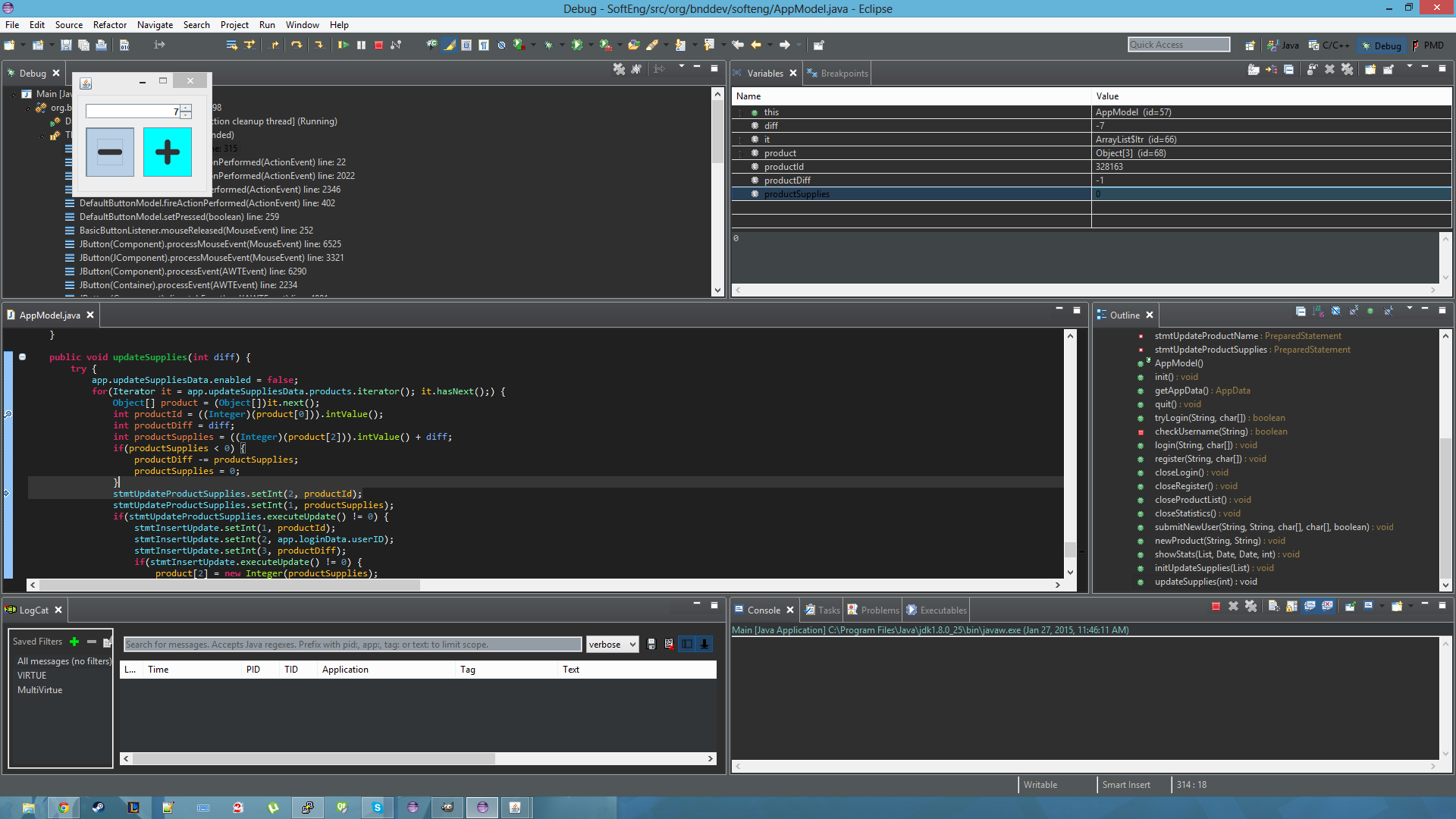
Παύση εφαρμογής στο breakpoint



Step Over για να δούμε τις μεταβλητές productDiff / productSupplies



Επιβεβαίωση μετά το step over ότι οι μεταβλητές είναι σωστές

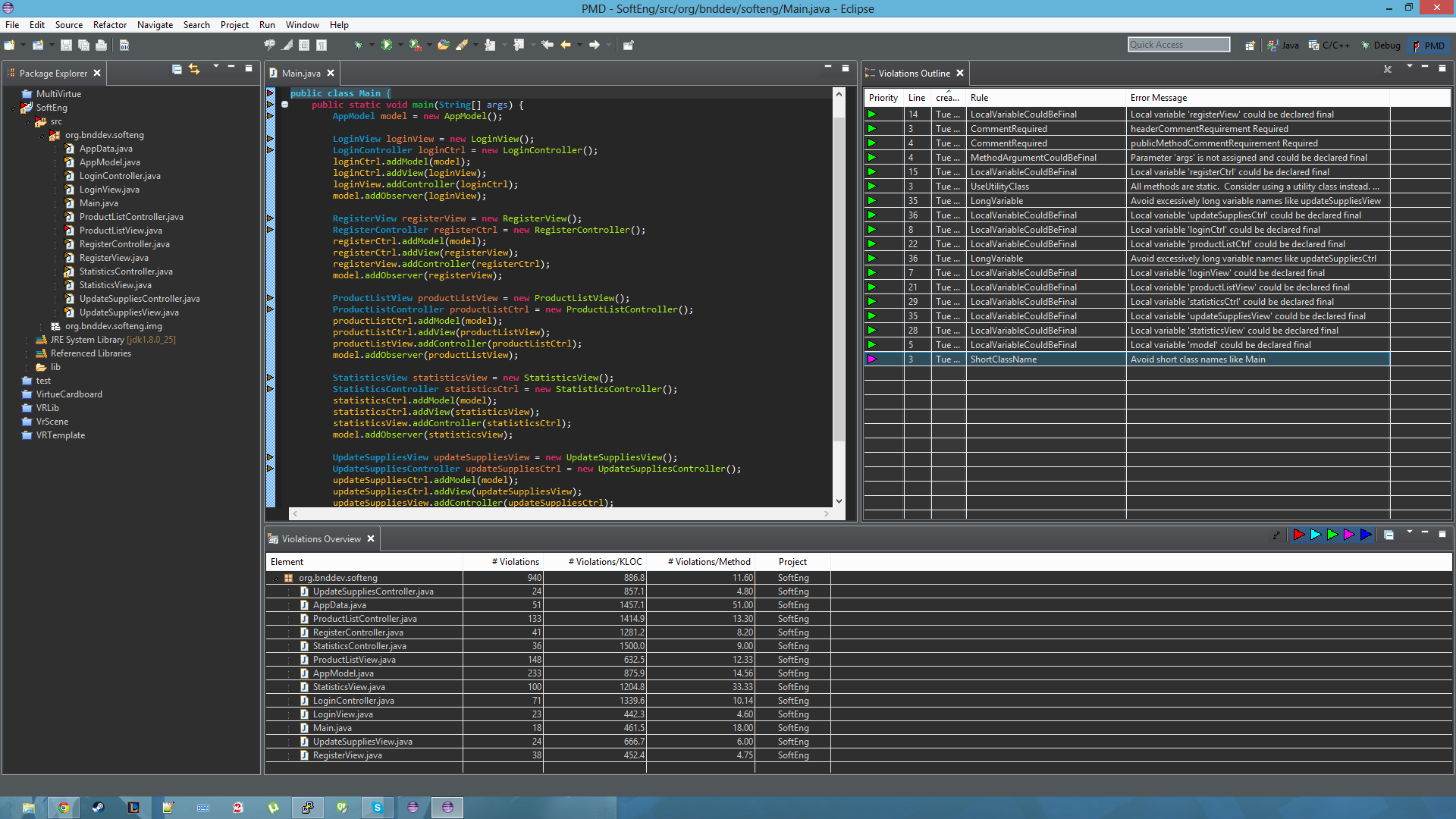


**ΑΠΟΤΕΛΕΣΜΑΤΑ STATIC ANALYSIS**

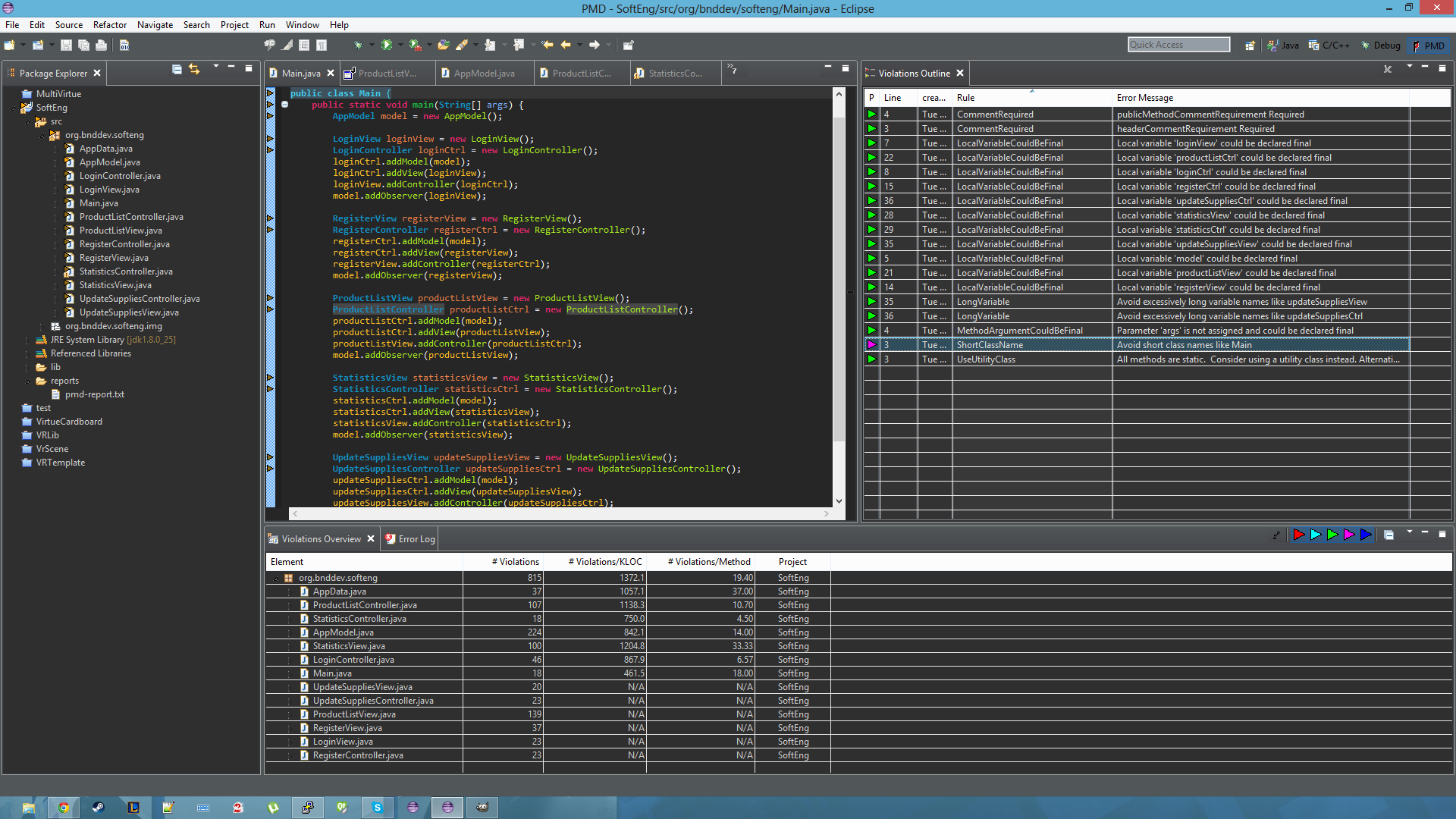
Η στατική ανάλυση κατά κάποιο τρόπο είναι μια πρώτη εκσφαλμάτωση του κώδικα χωρίς όμως αυτός να τρέξει. Τα εργαλεία στατικής ανάλυσης διαβάζουν τον κώδικα σαν text αρχείο και επισημαίνουν σφάλματα ή προτείνουν αλλαγές στον κώδικα για να γίνει πιο αποτελεσματικός και ασφαλείς , με την έννοια του να μην παρουσιαστούν πιθανά προβλήματα κατά την εκτέλεση του.

Μετά την στατική ανάλυση που τρέξαμε προέκυψαν δυο αρχεία PMD τα οποία είναι μεγάλα σε μέγεθος και δεν μπορούν να παρουσιαστούν στο παρών έγγραφο. Τα αρχεία είναι διαθέσιμα με τίτλους ***pmd-report-before*** και ***pmd-report-after*** , στον φάκελο της εργασίας και αναφέρονται στην έξοδο της στατικής ανάλυσης πριν και μετά την πρώτη φορά που την τρέξαμε. Η διαφορά, που ίσως να μην γίνεται εύκολα αντιληπτή, είναι πως στο *pmd-report-before* έχουμε κάποια παραπάνω σφάλματα που διορθώθηκαν και δεν υπάρχουν στο *pmd-report-after.*

PMD before

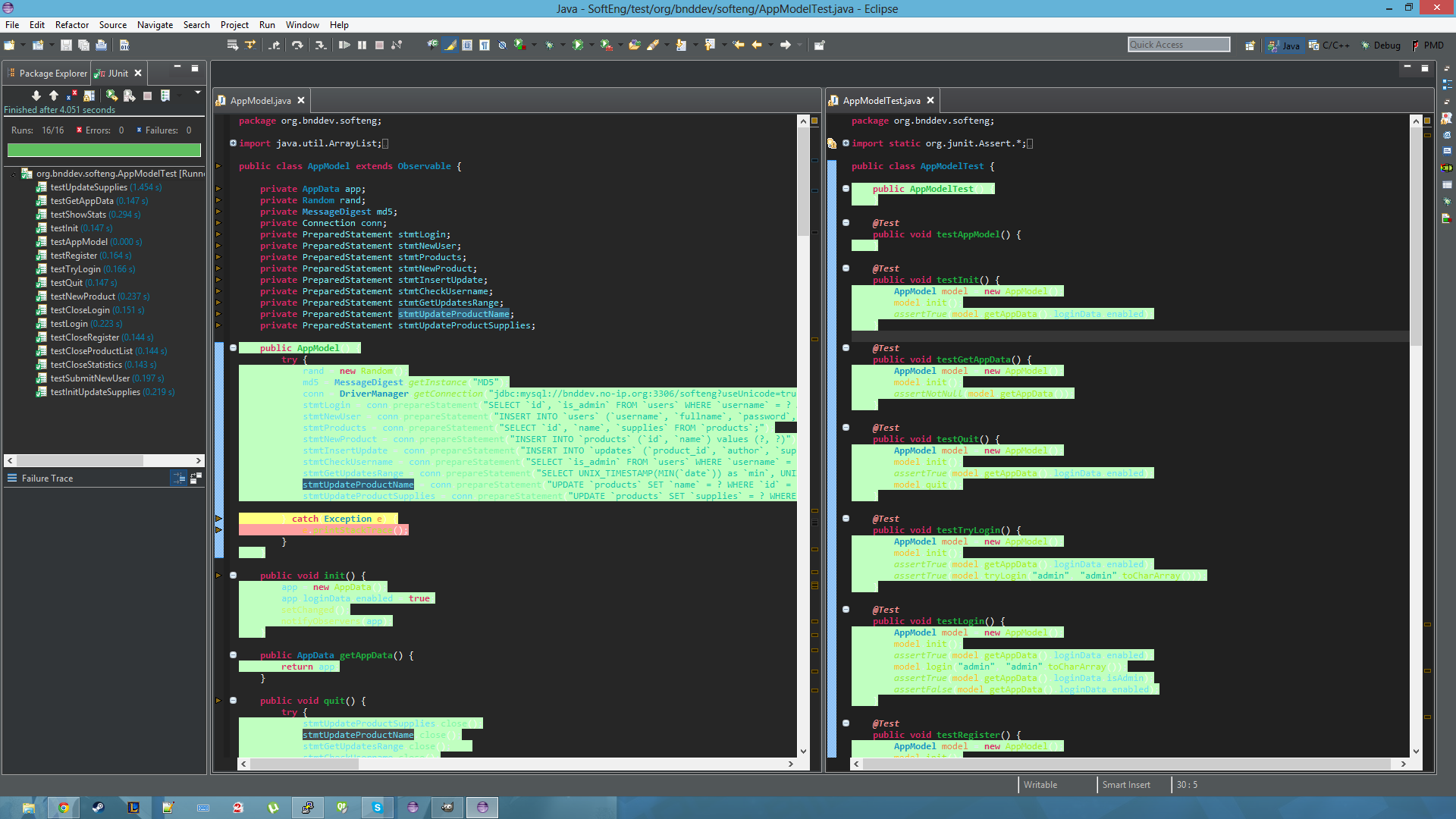


PMD after



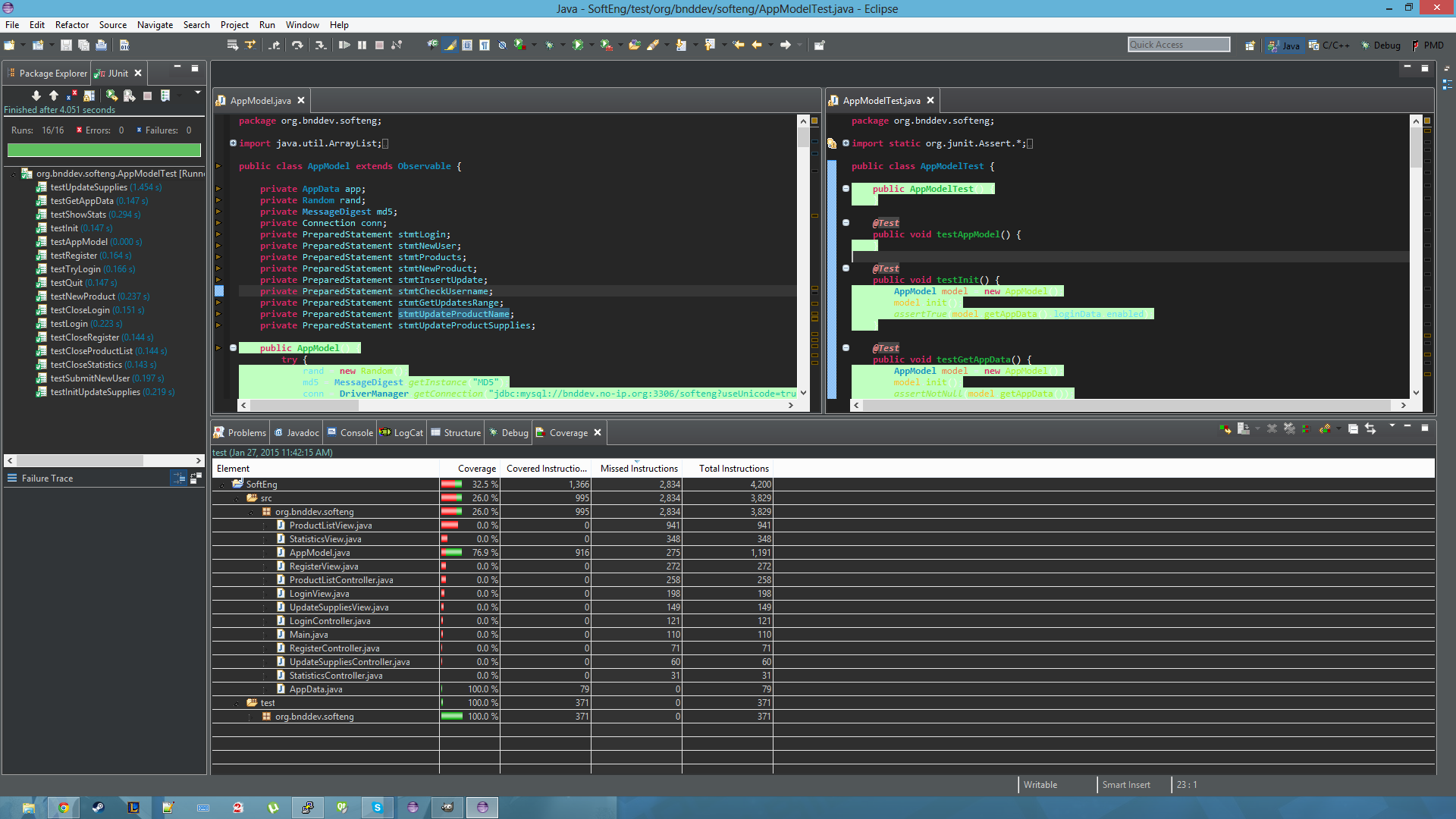
**UNIT TESTING**

Το unit testing είναι η διαδικασία κατά την οποία θεωρούμε τον κώδικα σαν ένα σύνολο από units ( κομμάτια κώδικα , πχ κλάσεις , μεθόδους κτλ. ) και συνεπώς τεστάρουμε κάθε unit για σφάλματα , ορίζοντας ένα σύνολο από ελέγχους σχετικούς με την κλάση που θέλουμε να γίνουν. Στο παράδειγμά μας κάνουμε unit testing στην κλάση που αφορά την σύνδεση του χρήστη στο σύστημα. Με τον τρόπο αυτό ελέγχουμε τα διάφορα σενάρια σύνδεσης του χρήστη στο σύστημα.



**Code Coverage**

Το code coverage μας δείχνει το πόσο της εκατό του κώδικά μας έχει καλυφτεί από την διαδικασία της εκσφαλμάτωσης. Πολλές φορές υπάρχουν κομμάτια κώδικα μέσα σε ένα πρόγραμμα που μπορεί να εκτελούνται πολύ σπάνια ή και καθόλου λόγο των εντολών συνθήκης που δεν το επιτρέπουν. Κάτι τέτοιο ωστόσο είναι επίφοβο , μιας και σε πραγματικές συνθήκες μπορεί να εκτελεστεί αυτό το κομμάτι κώδικα και να έχουμε πρόβλημα αν περιέχει σφάλματα. Αυτό ακριβώς έρχεται να απορρίψει ένα tool του Unit testing που μας βοηθάει να εκσφαλματώσουμε και περιοχές του κώδικα που δεν είναι συχνά προσπελάσιμες. Για το πρόγραμμά μας το ποσοστό του κώδικα που καλύφτηκε είναι 32.5% .



**USER ACCEPTANCE TEST**

To user acceptance test είναι ένα black box testing μιας και αυτός που το εκτελεί δεν βλέπει τι συμβαίνει στον κώδικα , αλλά τον ενδιαφέρει μόνο η σωστή συμπεριφορά του προγράμματος σε διάφορα σενάρια ελέγχου. Η πιο κλασική μορφή black box testing είναι αυτή κατά την οποία ο χρήστης αλληλεπιδρά με τα διάφορα views του προγράμματος ( buttons , textboxes , radio buttons κτλ.) και τεστάρει αν αυτά συμπεριφέρονται όπως θα περίμενε καθώς τους αλλάζει τις τιμές.

Για το παράδειγμα μας το User Acceptance Test είναι διαθέσιμο και πάλι σε εξωτερικό αρχείο μέσα στον φάκελο της εργασίας μας , με τίτλο ***UAT.***

**ΥΠΟΣΕΙΜΗΩΣΕΙΣ**

1. Επειδή τα διάφορα screenshots και οι εικόνες που παρατίθενται είναι σε μικρή ανάλυση , για να τις δείτε καλύτερα υπάρχουν ξεχωριστά , στον φάκελο με όνομα ***Screenshots*** στον φάκελο της εργασίας.
2. Οι κωδικοί για login στο πρόγραμμά μας ( username/password)είναι για τον :

administrator : admin/admin

υπάλληλο : Gmath/2117