|  |
| --- |
| **Σύστημα Έκδοσης Θεατρικών Εισιτηρίων** |
| **Μηχανική Λογισμικού** |
| Ακαδημαϊκό Έτος 2013-2014  [Χειμ. Εξάμηνο] | |
| Πιλάτη Ελένη, Α. Μ.: 39579  Πιλάτη Ευαγγελία, Α. Μ.: 39578  Αφεντάκη Λυδία, Α. Μ.: 39353 |
|  |

Περιεχόμενα

[1. Εισαγωγή 5](#_Toc378354554)

[2. Ανάλυση των απαιτήσεων του συστήματος 6](#_Toc378354555)

[2.1 Περιγραφή των εμπλεκομένων ρόλων του συστήματος 6](#_Toc378354556)

[2.2 Συνοπτική Περιγραφή των Περιπτώσεων Χρήσης 7](#_Toc378354557)

[2.3 UML Διάγραμμα Περιπτώσεων Χρήσης 9](#_Toc378354558)

[2.4 Ανάλυση Λειτουργικών Απαιτήσεων 10](#_Toc378354559)

[3. Διαγραμματική Απεικόνιση του Συστήματος (UML Διαγράμματα) 23](#_Toc378354560)

[3.1 UML Διάγραμμα Κλάσεων 23](#_Toc378354561)

[3.2 UML Διαγράμματα Ακολουθίας 24](#_Toc378354562)

[3.3 UML Διαγράμματα Δραστηριοτήτων 26](#_Toc378354563)

[4. Unit-testing 28](#_Toc378354564)

[Παράρτημα Α’ – Πηγαίος Κώδικας 32](#_Toc378354565)

[Παράρτημα Β’ – Screenshots του Συστήματος 47](#_Toc378354566)

**Πίνακας Εικόνων**

[Εικόνα 1. UML Διάγραμμα Περιπτώσεων Χρήσης 9](#_Toc378354567)

[Εικόνα 2. UML Διάγραμμα κλάσεων 23](#_Toc378354568)

[Εικόνα 3. UML Διάγραμμα Ακολουθίας για τον Ταμία 24](#_Toc378354569)

[Εικόνα 4. UML Διάγραμμα Ακολουθίας για τον Διαχειριστή για την εμφάνιση των διαθέσιμων/δεσμευμένων θέσεων 24](#_Toc378354570)

[Εικόνα 5. UML Διάγραμμα ακολουθίας για τον Διαχειριστή για την τροποποίηση μιας παράστασης 25](#_Toc378354571)

[Εικόνα 6. UML Διάγραμμα Δραστηριοτήτων για τον Ταμία 26](#_Toc378354572)

[Εικόνα 7. UML Διάγραμμα δραστηριοτήτων για τον Διαχειριστή του συστήματος 27](#_Toc378354573)

[Εικόνα 8. Είσοδος στο Σύστημα ως Διαχειριστής 47](#_Toc378354574)

[Εικόνα 9. Είσοδος στο Σύστημα ως Ταμίας 47](#_Toc378354575)

[Εικόνα 10. Εισαγωγή λάθους κωδικού 48](#_Toc378354576)

[Εικόνα 11. Επιλογή παράστασης 48](#_Toc378354577)

[Εικόνα 12. Επιλογή ώρας παράστασης 49](#_Toc378354578)

[Εικόνα 13. Επιλογή θέσης 49](#_Toc378354579)

[Εικόνα 14. Εισαγωγή νέας παράστασης 49](#_Toc378354580)

**Ευρετήριο Πινάκων**

[Πίνακας 1. Κατάλογος Περιπτώσεων Χρήσης 10](#_Toc378354582)

[Πίνακας 2. UC-1. Είσοδος στο Σύστημα 11](#_Toc378354583)

[Πίνακας 3. UC-2. Επιλογή Παράστασης 12](#_Toc378354584)

[Πίνακας 4. UC-3. Επιλογή Ώρας 13](#_Toc378354585)

[Πίνακας 5. UC-4. Εμφάνιση Διαθέσιμων Θέσεων 14](#_Toc378354586)

[Πίνακας 6. UC-5. Επιλογή Θέσης (-ων) 15](#_Toc378354587)

[Πίνακας 7. UC-6. Ανασκόπηση Κράτησης 16](#_Toc378354588)

[Πίνακας 8. UC-7. Τροποποίηση Κράτησης 17](#_Toc378354589)

[Πίνακας 9. UC-8. Κράτηση Εισιτηρίου(-ων) 18](#_Toc378354590)

[Πίνακας 10. UC-9. Εκτύπωση Εισιτηρίου(-ων) 19](#_Toc378354591)

[Πίνακας 11. UC-10. Καταχώρηση νέας παράστασης (-ων) 20](#_Toc378354592)

[Πίνακας 12. UC-11. Τροποποίηση υπάρχουσας παράστασης (-ων) 21](#_Toc378354593)

[Πίνακας 13. UC-12. Διαγραφή παράστασης (-ων) 22](#_Toc378354594)

# Εισαγωγή

Σκοπός της παρούσας εργασίας είναι η δημιουργία ενός πρωτοτύπου για ένα σύστημα έκδοσης θεατρικών εισιτηρίων. Το σύστημα αυτό προβλέπεται να χρησιμοποιηθεί σε ένα μικρό θέατρο, και μετά την υιοθέτησή του στόχος είναι η επίβλεψη και ο έλεγχος των παραστάσεων που παίζονται εκείνη την χρονική στιγμή στο θέατρο. Πιο συγκεκριμένα, μέσω του συστήματος, ο ταμίας θα έχει την δυνατότητα να επιλέγει τις διαθέσιμες παραστάσεις και ώρες παρουσίασης και αντίστοιχα να βλέπει τις διαθέσιμες ελεύθερες θέσεις και να προτείνει στους θεατές θέσεις που ανταποκρίνονται στις επιθυμίες τους. Κατά τη σχεδίαση της διεπαφής του συστήματος, στόχος είναι η ευχρηστία και ο περιορισμός, όσο αυτό είναι δυνατόν, του χρόνου εκμάθησης του συστήματος από τους χρήστες. Υπάρχουν δύο κατηγορίες χρηστών: ο Διαχειριστής του συστήματος (admin) και ο Ταμίας (cashier).

Μέσω της παρούσας εργασίας θα γίνει σε πρώτη φάση καταγραφή και ανάλυση των απαιτήσεων του συστήματος. Αυτό θα πραγματοποιηθεί μέσω της ανάλυσης των περιπτώσεων χρήσης του συστήματος, τόσο σε επίπεδο απεικόνισης μέσω του UML διαγράμματος περιπτώσεων χρήσης (UML Use Case Diagram), όσο και σε επίπεδο λεκτικής περιγραφής. Στη συνέχεια, θα γίνει περιγραφή των μη-λειτουργικών απαιτήσεων του συστήματος και των απαιτήσεων για τις εξωτερικές διεπαφές.

Κατά τη σχεδίαση του συστήματος, θα δοθεί το UML διάγραμμα κλάσεων (UML Class Diagram), το οποίο θα προκύψει μέσω της τεχνικής reverse engineering από τον κώδικα της εφαρμογής. Για την περιγραφή της συμπεριφοράς του συστήματος, θα δοθούν τα διαγράμματα ακολουθίας (UML Sequence Diagrams) και τα διαγράμματα δραστηριοτήτων (UML Activity Diagrams). Τέλος, θα δοθούν τα αποτελέσματα της static analysis πριν και μετά και του unit testing, καθώς και των σεναρίων για user acceptance testing.

Στο παράρτημα της εργασίας, βρίσκονται screenshots του συστήματος, καθώς και ο πηγαίος κώδικας του συστήματος.

# Ανάλυση των απαιτήσεων του συστήματος

## Περιγραφή των εμπλεκομένων ρόλων του συστήματος

Υπάρχουν δύο κατηγορίες χρηστών που θα χρησιμοποιήσουν το Σύστημα Έκδοσης Θεατρικών Εισιτηρίων. Αυτές οι κατηγορίες είναι ο Διαχειριστής του Συστήματος (System Administrator) και ο Ταμίας του θεάτρου (Cashier).

1. ***Διαχειριστής (System Administrator)*:** Ο διαχειριστής του συστήματος θα έχει την δυνατότητα:
   1. *Να δημιουργεί και να καταχωρεί μια νέα θεατρική παράσταση*, καταχωρώντας πληροφορίες για αυτήν, καθώς και τις ώρες που θα παίζεται.
   2. *Να αλλάζει τα δεδομένα τα οποία είναι καταχωρημένα για τις υπάρχουσες παραστάσεις* (π.χ. καταχώρηση νέας ώρας, αλλαγή στο όνομα της παράστασης).
   3. *Να διαγράφει μια παράσταση*, καθώς και όλα τα δεδομένα τα οποία έχουν καταχωρισθεί για αυτήν.
   4. *Να έχει μια συνολική εικόνα για τις δεσμευμένες και τις ελεύθερες θέσεις μιας παράστασης μια συγκεκριμένη ημέρα και ώρα.* Μέσω αναφοράς που θα βγάζει το σύστημα, ο διαχειριστής του συστήματος θα πρέπει να έχει την δυνατότητα να έχει μια ξεκάθαρη εικόνα ανά πάσα στιγμή για τη διαχείριση του θεάτρου.
2. ***Ταμίας του θεάτρου (Cashier)*:** Ο ταμίας του θεάτρου μέσω του συστήματος θα έχει τη δυνατότητα:
   1. *Να επιλέγει τις διαθέσιμες παραστάσεις και την ώρα που αυτές παίζονται και να ενημερώνει τον θεατή για τις διαθέσιμες ελεύθερες θέσεις*.
   2. *Να κάνει κράτηση μιας ή παραπάνω θέσεων για μια συγκεκριμένη παράσταση μια συγκεκριμένη ώρα*.
   3. *Να εκτυπώνει το εισιτήριο για μια παράσταση.*

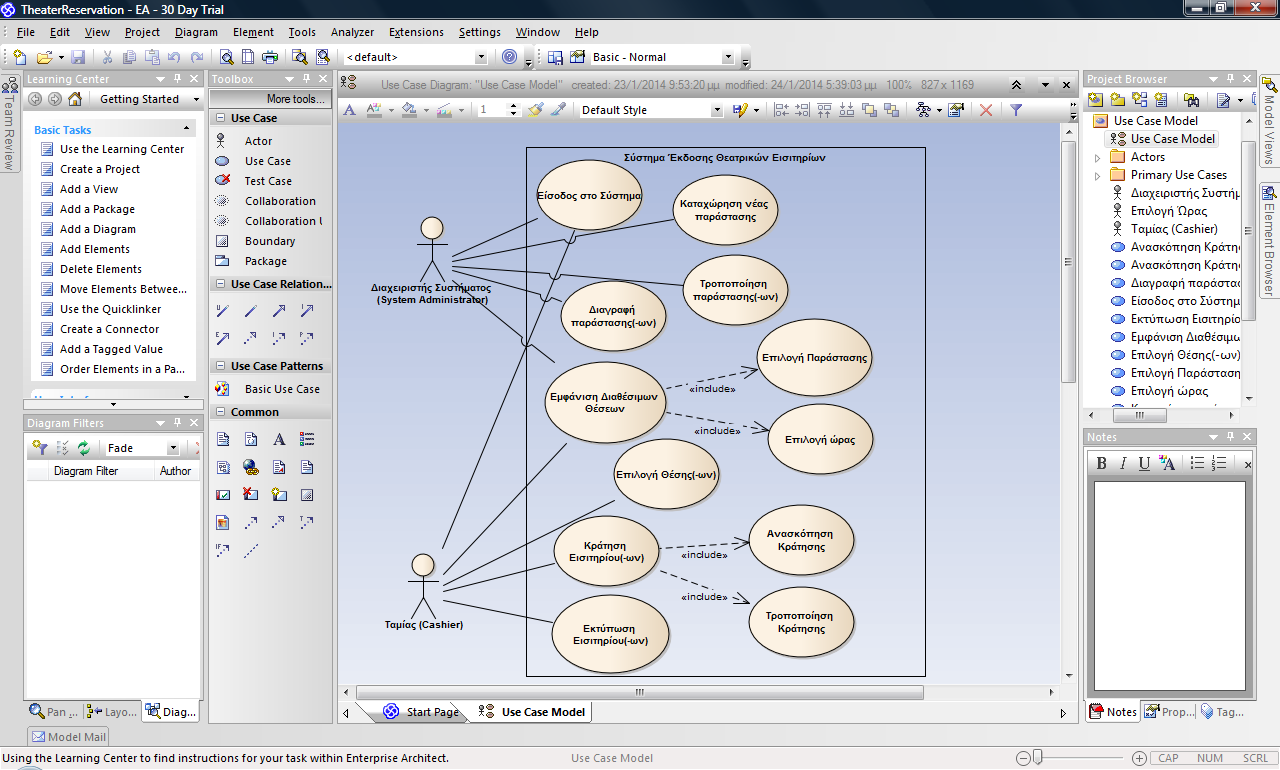
## Συνοπτική Περιγραφή των Περιπτώσεων Χρήσης

Οι περιπτώσεις χρήσης οι οποίες παρέχονται από το σύστημα είναι οι εξής:

1. ***Περίπτωση Χρήσης: Είσοδος στο Σύστημα.*** Ο χρήστης του συστήματος, είτε πρόκειται για τον Διαχειριστή του συστήματος είτε πρόκειται για τον Ταμία του θεάτρου εισέρχεται στο σύστημα εισάγοντας έναν κωδικό πρόσβασης. Ο κωδικός πρόσβασης για τον Διαχειριστή του συστήματος είναι «admin» και για τον Ταμία του θεάτρου «cashier». Με αυτό τον τρόπο ο χρήστης του συστήματος πιστοποιεί την ταυτότητα του και αποκτά την αντίστοιχη πρόσβαση και δικαιώματα στο σύστημα.
2. ***Περίπτωση Χρήσης: Επιλογή Παράστασης.*** Ο Ταμίας, αφού εισέλθει στο σύστημα, επιλέγει την παράσταση για την οποία επιθυμεί να λάβει πληροφορίες ή να κάνει κράτηση. Αντίστοιχα, και ο Διαχειριστής του συστήματος, αφού εισέλθει στο σύστημα, επιλέγει την παράσταση για την οποία θέλει να αποκτήσει πληροφορίες.
3. ***Περίπτωση Χρήσης: Επιλογή Ώρας.*** Ο Ταμίας, αφού εισέλθει στο σύστημα, επιλέγει την ώρα της παράστασης για την οποία επιθυμεί να λάβει πληροφορίες ή να κάνει κράτηση. Αντίστοιχα, και ο Διαχειριστής του συστήματος, αφού εισέλθει στο σύστημα, επιλέγει την ώρα της παράστασης για την οποία θέλει να αποκτήσει πληροφορίες.
4. ***Περίπτωση Χρήσης:*** ***Εμφάνιση Διαθέσιμων Θέσεων***. Ο Ταμίας, αφού εισέλθει στο σύστημα, μπορεί να δει τις διαθέσιμες θέσεις για μια συγκεκριμένη παράσταση μια συγκεκριμένη ώρα. Αντίστοιχα, και ο Διαχειριστής του συστήματος, αφού εισέλθει στο σύστημα, μπορεί μέσω της εμφάνισης των διαθέσιμων θέσεων για μια συγκεκριμένη παράσταση μια συγκεκριμένη ώρα να αποκτήσει τις πληροφορίες που χρειάζεται.
5. ***Περίπτωση Χρήσης: Επιλογή Θέσης (-ων).*** Σε αυτό το σημείο, θα πρέπει να διευκρινίσουμε ότι κράτηση θέσης μπορεί να κάνει μόνο ο Ταμίας. Συνεπώς, αυτή η περίπτωση χρήσης είναι διαθέσιμη μόνο για αυτόν. Ο ταμίας, αφού εισέλθει στο σύστημα, επιλέγει την θέση ή τις θέσεις για τις οποίες επιθυμεί ο θεατής να κάνει κράτηση για μια συγκεκριμένη παράσταση.
6. ***Περίπτωση Χρήσης:*** ***Ανασκόπηση κράτησης***. Αυτή η περίπτωση χρήσης είναι διαθέσιμη μόνο για τον Ταμία του θεάτρου, ο οποίος μετά την επιλογή της παράστασης, της ώρας και της θέσης ή των θέσεων έχει την δυνατότητα πριν την πραγματοποίηση της κράτησης να κάνει μια ανασκόπηση. Με αυτό τον τρόπο, έχει την δυνατότητα σε επικοινωνία με τον θεατή να κάνει έναν έλεγχο πριν την πραγματοποίηση της κράτησης.
7. ***Περίπτωση Χρήσης:*** ***Τροποποίηση κράτησης***. Αυτή η περίπτωση χρήσης είναι διαθέσιμη μόνο για τον Ταμία του θεάτρου. Σε περίπτωση που μετά την ανασκόπηση της κράτησης προέκυψε κάποια σφάλμα, ο Ταμίας έχει την δυνατότητα να κάνει αλλαγές, τόσο στην παράσταση, όσο στην ώρα και στην θέση (-εις).
8. ***Περίπτωση Χρήσης:*** ***Κράτηση Εισιτηρίου (-ων)***. Η κράτηση θέσης μπορεί να πραγματοποιηθεί μόνο από τον Ταμία του Θεάτρου. Μέσω αυτής της περίπτωσης χρήσης γίνεται η κράτηση του εισιτηρίου και κατ’ επέκταση της θέσης, έτσι ώστε αυτή να μην είναι διαθέσιμη για επόμενες κρατήσεις. Η εκτύπωση και η αγορά του εισιτηρίου μπορεί να γίνει σε επόμενη φάση.
9. ***Περίπτωση Χρήσης:*** ***Εκτύπωση Εισιτηρίου (-ων)***. Αυτή η περίπτωση χρήσης παρέχεται μόνο στον Ταμία του θεάτρου ο οποίος μετά την κράτηση του εισιτηρίου μπορεί να προχωρήσει στην εκτύπωσή του.
10. ***Περίπτωση Χρήσης:*** ***Καταχώρηση νέας παράστασης (-ων)***. Αυτή η περίπτωση χρήσης παρέχεται μόνο στο Διαχειριστή του συστήματος ο οποίος έχει την δυνατότητα να εισάγει τα δεδομένα που αφορούν μια νέα παράσταση που θα παίζεται στο θέατρο.
11. ***Περίπτωση Χρήσης:*** ***Τροποποίηση υπάρχουσας παράστασης (-ων)***. Αυτή η περίπτωση χρήσης παρέχεται μόνο στο Διαχειριστή του συστήματος ο οποίος έχει την δυνατότητα να τροποποιήσει τα δεδομένα τα οποία αφορούν μια παράσταση, όπως το όνομα της παράστασης και την ώρα στην οποία παίζεται η παράσταση.
12. ***Περίπτωση Χρήσης:*** ***Διαγραφή παράστασης (-ων)***. Αυτή η περίπτωση χρήσης παρέχεται μόνο στο Διαχειριστή του συστήματος ο οποίος έχει την δυνατότητα να διαγράψει μια υπάρχουσα παράσταση καθώς και τα δεδομένα τα οποία είναι αποθηκευμένα για αυτήν.

## UML Διάγραμμα Περιπτώσεων Χρήσης

Στην παρακάτω εικόνα δίνεται το UML διάγραμμα περιπτώσεων χρήσης του συστήματος.



Εικόνα 1. UML Διάγραμμα Περιπτώσεων Χρήσης

## Ανάλυση Λειτουργικών Απαιτήσεων

Στον παρακάτω πίνακα δίνεται ο κατάλογος των περιπτώσεων χρήσης του συστήματος.

Πίνακας 1. Κατάλογος Περιπτώσεων Χρήσης

|  |  |
| --- | --- |
| Εμπλεκόμενοι Ρόλοι | Περιπτώσεις Χρήσης |
| Ταμίας  Διαχειριστής | UC-1. Είσοδος στο Σύστημα |
| UC-2. Επιλογή Παράστασης |
| UC-3. Επιλογή Ώρας |
| UC-4. Εμφάνιση Διαθέσιμων Θέσεων |
| Ταμίας | UC-5. Επιλογή Θέσης (-ων) |
| UC-6. Ανασκόπηση κράτησης |
| UC-7. Τροποποίηση κράτησης |
| UC-8. Κράτηση Εισιτηρίου (-ων) |
| UC-9. Εκτύπωση Εισιτηρίου (-ων) |
| Διαχειριστής | UC-10. Καταχώρηση νέας παράστασης (-ων) |
| UC-11. Τροποποίηση υπάρχουσας παράστασης (-ων) |
| UC-12. Διαγραφή παράστασης (-ων) |

Πίνακας . UC-1. Είσοδος στο Σύστημα

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-1 | | |
| Ονομασία: | | Είσοδος στο Σύστημα | | |
| Δημιουργήθηκε από: | Πιλάτη Ελένη | | Τελευταία ενημέρωση από: | Πιλάτη Ελένη |
| Ημερομηνία Συγγραφής: | 08/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 09/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας, Διαχειριστής του συστήματος | | |
| Περιγραφή: | | Ο Ταμίας/Διαχειριστής του συστήματος μπορεί να εισέλθει στο σύστημα εκτελώντας την εφαρμογή και εισάγοντας τον κωδικό του. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας/Διαχειριστής εκτελεί την εφαρμογή. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας/Διαχειριστής κάνει την επιλογή του για την είσοδό του στο σύστημα ως Ταμίας/Διαχειριστής. 2. Ο Ταμίας/Διαχειριστής εισάγει έναν έγκυρο κωδικό. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας/Διαχειριστής εισέρχεται στο σύστημα. 2. Εμφανίζεται ένα μενού με τις διαθέσιμες ενέργειες που μπορεί να εκτελέσει. | | |
| Φυσιολογική Ροή: | | 1. Ο Ταμίας/Διαχειριστής επιλέγει τον ρόλο με βάσει τον οποίο θα εισέλθει στο σύστημα (ως ταμίας ή ως διαχειριστής). 2. Ο Ταμίας/Διαχειριστής εισάγει τον κωδικό του σε ένα πεδίο κειμένου. 3. Το σύστημα ελέγχει εάν ο κωδικός είναι έγκυρος με βάση τον ρόλο που έχει επιλέξει ο χρήστης (ταμίας ή διαχειριστής). 4. Το σύστημα εμφανίζει ένα μενού με τις διαθέσιμες ενέργειες. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | 1. Ο Ταμίας/Διαχειριστής εισάγει λάθος κωδικό. 2. Το σύστημα ενημερώνει τον χρήστη για τον μη έγκυρο κωδικό. 3. Ο Ταμίας/Διαχειριστής επανεισάγει τον κωδικό. | | |
| Ενσωματώνει: | | - | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 200 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 3. UC-2. Επιλογή Παράστασης

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-2 | | |
| Ονομασία: | | Επιλογή Παράστασης | | |
| Δημιουργήθηκε από: | Πιλάτη Ευαγγελία | | Τελευταία ενημέρωση από: | Αφεντάκη Λυδία |
| Ημερομηνία Συγγραφής: | 09/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 11/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας, Διαχειριστής του συστήματος | | |
| Περιγραφή: | | Ο Ταμίας/Διαχειριστής του συστήματος μπορεί να επιλέξει μια παράσταση από μια λίστα με όλες τις διαθέσιμες παραστάσεις. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας/Διαχειριστής εισέρχεται στο σύστημα. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας/Διαχειριστής εισέρχεται στο σύστημα. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας/Διαχειριστής επιλέγει την παράσταση για την οποία θέλει να λάβει πληροφορίες ή να κάνει κράτηση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. Το σύστημα εμφανίζει όλες τις τρέχουσες παραστάσεις. 3. Ο Ταμίας/Διαχειριστής επιλέγει μια παράσταση. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 200 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 4. UC-3. Επιλογή Ώρας

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-3 | | |
| Ονομασία: | | Επιλογή Ώρας | | |
| Δημιουργήθηκε από: | Αφεντάκη Λυδία | | Τελευταία ενημέρωση από: | Αφεντάκη Λυδία |
| Ημερομηνία Συγγραφής: | 10/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 12/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας, Διαχειριστής του συστήματος | | |
| Περιγραφή: | | Ο Ταμίας/Διαχειριστής του συστήματος μπορεί να επιλέξει την ώρα την οποία παίζεται η θεατρική παράσταση από μια λίστα με όλες τις διαθέσιμες ώρες εκτέλεσης της παράστασης. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας/Διαχειριστής επιλέγει μια παράσταση. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας/Διαχειριστής εισέρχεται στο σύστημα. 2. Ο Ταμίας/Διαχειριστής έχει επιλέξει μια παράσταση. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας/Διαχειριστής επιλέγει την ώρα εκτέλεσης της θεατρικής παράστασης για την οποία θέλει να λάβει πληροφορίες ή να κάνει κράτηση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. Το σύστημα εμφανίζει όλες τις τρέχουσες ώρες. 4. Ο Ταμίας/Διαχειριστής επιλέγει μια ώρα εκτέλεση της θεατρικής παράστασης. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 200 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 5. UC-4. Εμφάνιση Διαθέσιμων Θέσεων

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-4 | | |
| Ονομασία: | | Εμφάνιση Διαθέσιμων Θέσεων | | |
| Δημιουργήθηκε από: | Πιλάτη Ευαγγελία | | Τελευταία ενημέρωση από: | Πιλάτη Ευαγγελία |
| Ημερομηνία Συγγραφής: | 11/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 13/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας, Διαχειριστής του συστήματος | | |
| Περιγραφή: | | Το σύστημα εμφανίζει όλες τις διαθέσιμες και δεσμευμένες θέσεις για την παράσταση και την ώρα που έχει επιλέξει ο χρήστης. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας/Διαχειριστής επιλέγει μια ώρα εκτέλεσης της παράστασης. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας/Διαχειριστής εισέρχεται στο σύστημα. 2. Ο Ταμίας/Διαχειριστής έχει επιλέξει μια παράσταση. 3. Ο Ταμίας/Διαχειριστής έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. | | |
| Τελική Κατάσταση: | | 1. Το σύστημα εμφανίζει όλες τις διαθέσιμες και δεσμευμένες θέσεις σε μια προεπισκόπηση του θέατρο. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. Το σύστημα εμφανίζει όλες τις διαθέσιμες και δεσμευμένες θέσεις του θεάτρου. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 200 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 6. UC-5. Επιλογή Θέσης (-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-5 | | |
| Ονομασία: | | Επιλογή Θέσης (-ων) | | |
| Δημιουργήθηκε από: | Πιλάτη Ελένη | | Τελευταία ενημέρωση από: | Πιλάτη Ελένη |
| Ημερομηνία Συγγραφής: | 13/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 15/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας | | |
| Περιγραφή: | | Ο Ταμίας έχοντας στην διάθεσή του την προεπισκόπηση των θέσεων όλου του θεάτρου και βλέποντας τις διαθέσιμες και τις δεσμευμένες θέσεις μπορεί να επιλέξει μια θέση(-εις) για την πραγματοποίηση μιας κράτησης. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας επιλέγει μια ώρα εκτέλεσης της παράστασης. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας εισέρχεται στο σύστημα. 2. Ο Ταμίας έχει επιλέξει μια παράσταση. 3. Ο Ταμίας έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. 4. Το σύστημα έχει εμφανίσει μια προεπισκόπηση του θεάτρου με τις δεσμευμένες και τις διαθέσιμες θέσεις. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας επιλέγει θέση(-εις) για κράτηση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. Ο Ταμίας επιλέγει μια ή περισσότερες θέσεις για κράτηση. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3, UC-4 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 150 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας . UC-6. Ανασκόπηση Κράτησης

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-6 | | |
| Ονομασία: | | Ανασκόπηση Κράτησης | | |
| Δημιουργήθηκε από: | Αφεντάκη Λυδία | | Τελευταία ενημέρωση από: | Πιλάτη Ευαγγελία |
| Ημερομηνία Συγγραφής: | 15/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 16/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας | | |
| Περιγραφή: | | Ο Ταμίας έχοντας επιλέξει μια ή περισσότερες θέσεις για κράτηση μπορεί να δει την ανασκόπηση της κράτησης πριν προχωρήσει στην κράτηση ενός ή περισσότερων εισιτηρίων. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας επιλέγει μια ή περισσότερες θέσεις για κράτηση. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας εισέρχεται στο σύστημα. 2. Ο Ταμίας έχει επιλέξει μια παράσταση. 3. Ο Ταμίας έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. 4. Ο Ταμίας έχει επιλέξει μια ή περισσότερες θέσεις για κράτηση. | | |
| Τελική Κατάσταση: | | 1. Το σύστημα εμφανίζει μια ανασκόπηση της κράτησης. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. Το σύστημα εμφανίζει την ανασκόπηση της κράτησης. 7. Ο Ταμίας βλέποντας την ανασκόπηση της κράτησης δεν επιθυμεί να κάνει κάποια αλλαγή. 8. UC-8 9. UC-9 | | |
| Εναλλακτική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. Το σύστημα εμφανίζει την ανασκόπηση της κράτησης. 7. Ο Ταμίας βλέποντας την ανασκόπηση της κράτησης επιθυμεί να κάνει κάποιες τροποποιήσεις. 8. UC-7 9. UC-8 10. UC-9 | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3, UC-4, UC-5 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 150 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 8. UC-7. Τροποποίηση Κράτησης

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-7 | | |
| Ονομασία: | | Τροποποίηση Κράτησης | | |
| Δημιουργήθηκε από: | Πιλάτη Ελένη | | Τελευταία ενημέρωση από: | Πιλάτη Ελένη |
| Ημερομηνία Συγγραφής: | 16/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 16/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας | | |
| Περιγραφή: | | Ο Ταμίας μετά την ανασκόπηση της κράτησης επιθυμεί να κάνει κάποιες τροποποιήσεις στην κράτηση. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας επιλέγει τροποποίηση της κράτησης. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας εισέρχεται στο σύστημα. 2. Ο Ταμίας έχει επιλέξει μια παράσταση. 3. Ο Ταμίας έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. 4. Ο Ταμίας έχει επιλέξει μια ή περισσότερες θέσεις για κράτηση. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας τροποποιεί την κράτηση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. Το σύστημα εμφανίζει την ανασκόπηση της κράτησης. 7. Ο Ταμίας βλέποντας την ανασκόπηση της κράτησης επιθυμεί να κάνει κάποιες τροποποιήσεις. 8. UC-8 9. UC-9 | | |
| Εναλλακτική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. Το σύστημα εμφανίζει την ανασκόπηση της κράτησης. 7. Ο Ταμίας βλέποντας την ανασκόπηση της κράτησης επιθυμεί να κάνει κάποιες τροποποιήσεις. 8. Κατά την είσοδό του στην φόρμα για την τροποποίηση της κράτησης, ο Ταμίας αποφασίζει ότι δεν θέλει να τροποποιήσει την κράτηση και ακυρώνει την τροποποίηση. 9. UC-8 10. UC-9 | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3, UC-4, UC-5 | | |
| Προτεραιότητα: | | 2 | | |
| Συχνότητα χρήσης: | | Περίπου 50 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 9. UC-8. Κράτηση Εισιτηρίου(-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-8 | | |
| Ονομασία: | | Κράτηση Εισιτηρίου(-ων) | | |
| Δημιουργήθηκε από: | Πιλάτη Ευαγγελία | | Τελευταία ενημέρωση από: | Πιλάτη Ευαγγελία |
| Ημερομηνία Συγγραφής: | 17/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 18/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας | | |
| Περιγραφή: | | Ο Ταμίας μετά την ανασκόπηση της κράτησης πραγματοποιεί κράτηση ενός ή περισσότερων εισιτηρίων. | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας επιλέγει κράτηση ενός ή περισσότερων εισιτηρίων. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας εισέρχεται στο σύστημα. 2. Ο Ταμίας έχει επιλέξει μια παράσταση. 3. Ο Ταμίας έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. 4. Ο Ταμίας έχει επιλέξει μια ή περισσότερες θέσεις για κράτηση. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας πραγματοποιεί την κράτηση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. UC-6 7. UC-8 8. UC-9 | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3, UC-4, UC-5, UC-6 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 150 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 10. UC-9. Εκτύπωση Εισιτηρίου(-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-9 | | |
| Ονομασία: | | Εκτύπωση Εισιτηρίου(-ων) | | |
| Δημιουργήθηκε από: | Αφεντάκη Λυδία | | Τελευταία ενημέρωση από: | Αφεντάκη Λυδία |
| Ημερομηνία Συγγραφής: | 19/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 21/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Ταμίας | | |
| Περιγραφή: | | Ο Ταμίας μετά την πραγματοποίηση της κράτησης, προχωρά στην εκτύπωση του εισιτηρίου (-ων). | | |
| Γεγονός Εκκίνησης: | | Ο Ταμίας επιλέγει εκτύπωση ενός ή περισσότερων εισιτηρίων. | | |
| Προϋποθέσεις: | | 1. Ο Ταμίας εισέρχεται στο σύστημα. 2. Ο Ταμίας έχει επιλέξει μια παράσταση. 3. Ο Ταμίας έχει επιλέξει μια ώρα εκτέλεσης της παράστασης. 4. Ο Ταμίας έχει επιλέξει μια ή περισσότερες θέσεις για κράτηση. | | |
| Τελική Κατάσταση: | | 1. Ο Ταμίας πραγματοποιεί την εκτύπωση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. UC-3 4. UC-4 5. UC-5 6. UC-6 7. UC-8 8. UC-9 | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2, UC-3, UC-4, UC-5, UC-6, UC-8 | | |
| Προτεραιότητα: | | 1 | | |
| Συχνότητα χρήσης: | | Περίπου 150 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 11. UC-10. Καταχώρηση νέας παράστασης (-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-10 | | |
| Ονομασία: | | Καταχώρηση νέας παράστασης (-ων) | | |
| Δημιουργήθηκε από: | Πιλάτη Ελένη | | Τελευταία ενημέρωση από: | Πιλάτη Ελένη |
| Ημερομηνία Συγγραφής: | 15/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 17/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Διαχειριστής | | |
| Περιγραφή: | | Ο Διαχειριστής μπορεί να καταχωρίσει μια νέα παράσταση η οποία θα παίζεται στο θέατρο. | | |
| Γεγονός Εκκίνησης: | | Ο Διαχειριστής επιλέγει την καταχώρηση μιας νέας παράστασης. | | |
| Προϋποθέσεις: | | 1. Ο Διαχειριστής εισέρχεται στο σύστημα. | | |
| Τελική Κατάσταση: | | 1. Ο Διαχειριστής καταχωρεί μια νέα παράσταση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. Ο Διαχειριστής επιλέγει την καταχώρηση μιας νέας παράστασης. 3. Ο Διαχειριστής εισάγει το όνομα της νέας παράστασης. 4. Ο Διαχειριστής εισάγει την ώρα ή τις ώρες στις οποίες θα παίζεται η νέα παράσταση. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1 | | |
| Προτεραιότητα: | | 3 | | |
| Συχνότητα χρήσης: | | Περίπου 2 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

Πίνακας 12. UC-11. Τροποποίηση υπάρχουσας παράστασης (-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-11 | | |
| Ονομασία: | | Τροποποίηση υπάρχουσας παράστασης (-ων) | | |
| Δημιουργήθηκε από: | Πιλάτη Ευαγγελία | | Τελευταία ενημέρωση από: | Πιλάτη Ευαγγελία |
| Ημερομηνία Συγγραφής: | 17/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 18/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Διαχειριστής | | |
| Περιγραφή: | | Ο Διαχειριστής μπορεί να τροποποιήσει τα στοιχεία μια υπάρχουσας παράστασης η οποία παίζεται στο θέατρο. | | |
| Γεγονός Εκκίνησης: | | Ο Διαχειριστής επιλέγει την τροποποίηση μιας παράστασης. | | |
| Προϋποθέσεις: | | 1. Ο Διαχειριστής εισέρχεται στο σύστημα. 2. Ο Διαχειριστής επιλέγει μια παράσταση. | | |
| Τελική Κατάσταση: | | 1. Ο Διαχειριστής τροποποιεί τα στοιχεία της παράστασης. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. Ο Διαχειριστής επιλέγει να τροποποιήσει την παράσταση. 4. Ο Διαχειριστής τροποποιεί το όνομα της παράστασης ή την ώρα(-ες) που παίζεται η παράσταση. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2 | | |
| Προτεραιότητα: | | 3 | | |
| Συχνότητα χρήσης: | | Περίπου 5 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

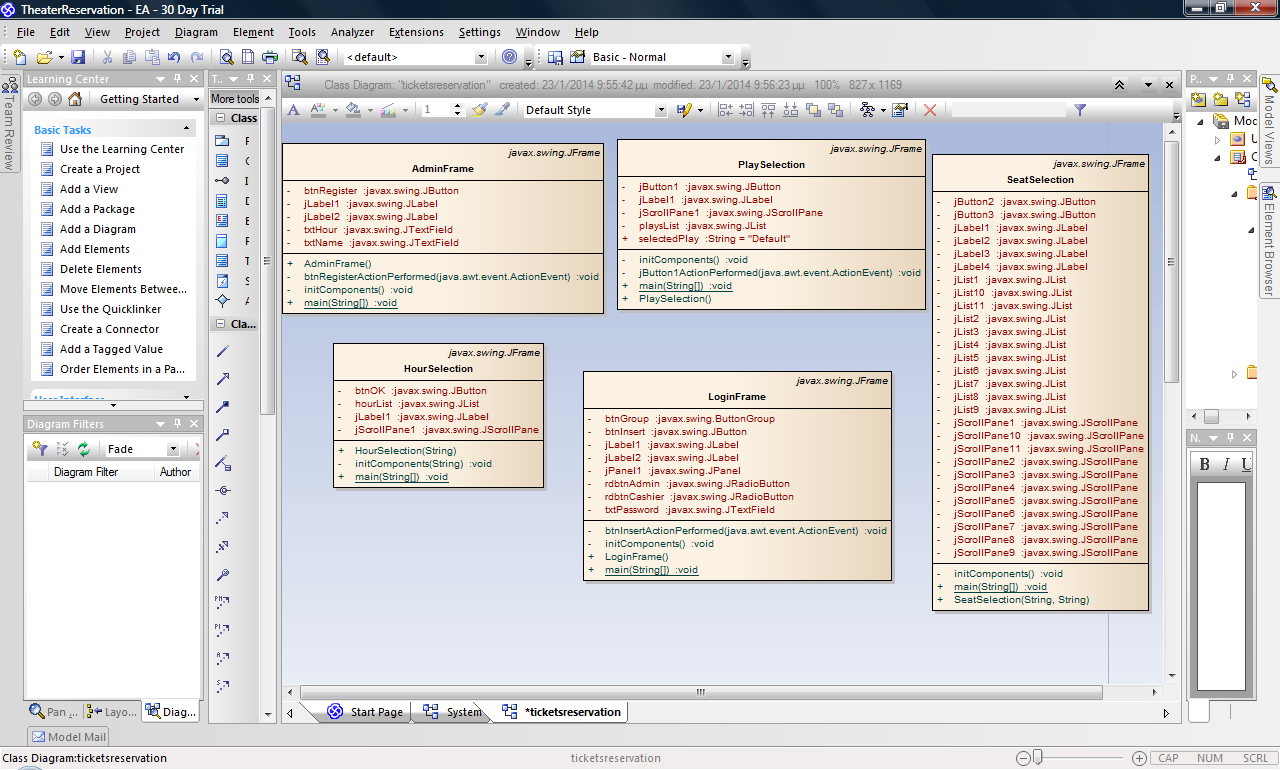
Πίνακας 13. UC-12. Διαγραφή παράστασης (-ων)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | | UC-12 | | |
| Ονομασία: | | Διαγραφή παράστασης (-ων) | | |
| Δημιουργήθηκε από: | Αφεντάκη Λυδία | | Τελευταία ενημέρωση από: | Αφεντάκη Λυδία |
| Ημερομηνία Συγγραφής: | 18/01/2014 | | Ημερομηνία τελευταίας ενημέρωσης: | 18/01/2014 |
| Εμπλεκόμενοι Ρόλοι: | | Διαχειριστής | | |
| Περιγραφή: | | Ο Διαχειριστής μπορεί να διαγράψει τα στοιχεία μια παράστασης που παίζεται στο θέατρο. | | |
| Γεγονός Εκκίνησης: | | Ο Διαχειριστής επιλέγει την τροποποίηση μιας παράστασης. | | |
| Προϋποθέσεις: | | 1. Ο Διαχειριστής εισέρχεται στο σύστημα. 2. Ο Διαχειριστής επιλέγει μια παράσταση. | | |
| Τελική Κατάσταση: | | 1. Ο Διαχειριστής διαγράφει μια παράσταση. | | |
| Φυσιολογική Ροή: | | 1. UC-1 2. UC-2 3. Ο Διαχειριστής επιλέγει να διαγράψει την παράσταση. 4. Το σύστημα εμφανίζει ένα μήνυμα επιβεβαίωσης της διαγραφής. 5. Ο Διαχειριστής απαντά θετικά στο μήνυμα επιβεβαίωσης και πραγματοποιείται η διαγραφή της παράστασης. | | |
| Εναλλακτική Ροή: | | - | | |
| Εξαιρέσεις: | | - | | |
| Ενσωματώνει: | | UC-1, UC-2 | | |
| Προτεραιότητα: | | 3 | | |
| Συχνότητα χρήσης: | | Περίπου 2 χρήσεις ανά ημέρα | | |
| Business Rules: | | - | | |
| Ειδικές απαιτήσεις: | | - | | |
| Υποθέσεις: | | - | | |
| Σημειώσεις και ζητήματα: | |  | | |

# Διαγραμματική Απεικόνιση του Συστήματος (UML Διαγράμματα)

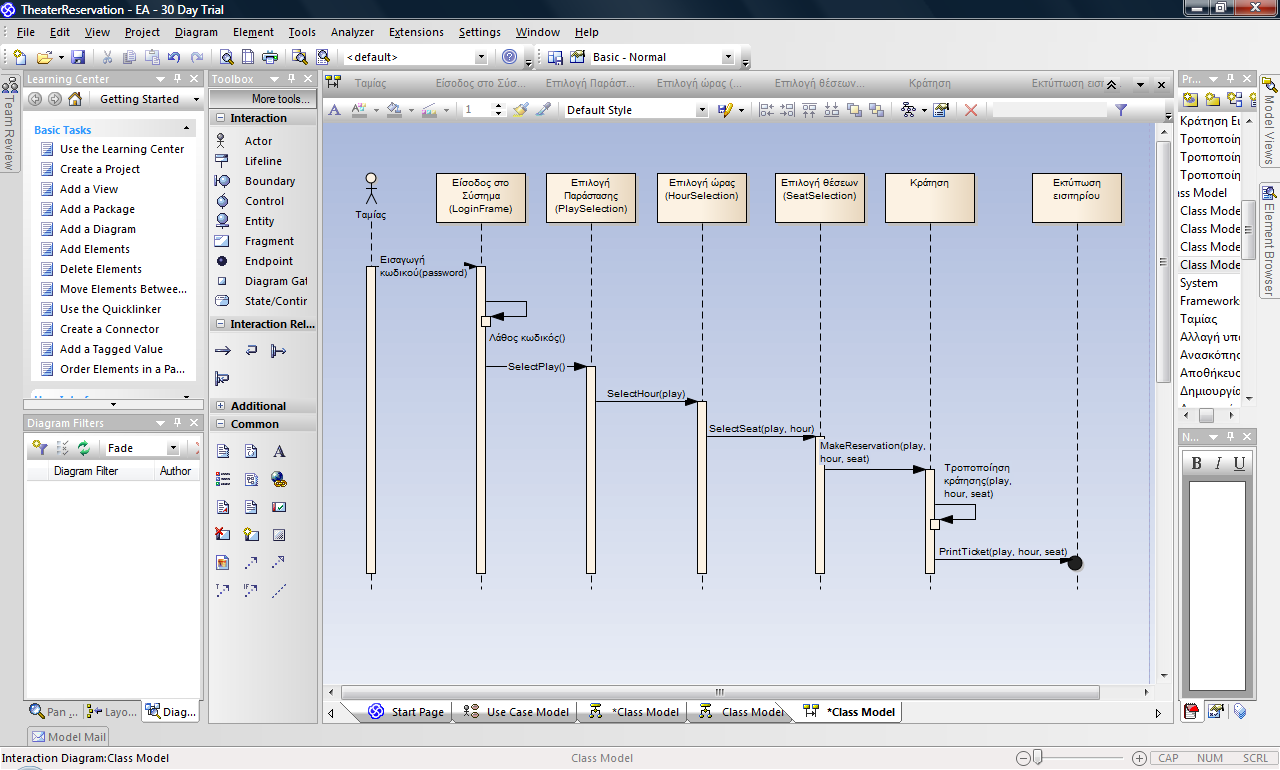
## UML Διάγραμμα Κλάσεων

Στην παρακάτω εικόνα απεικονίζεται το UML Διάγραμμα Κλάσεων του συστήματος.

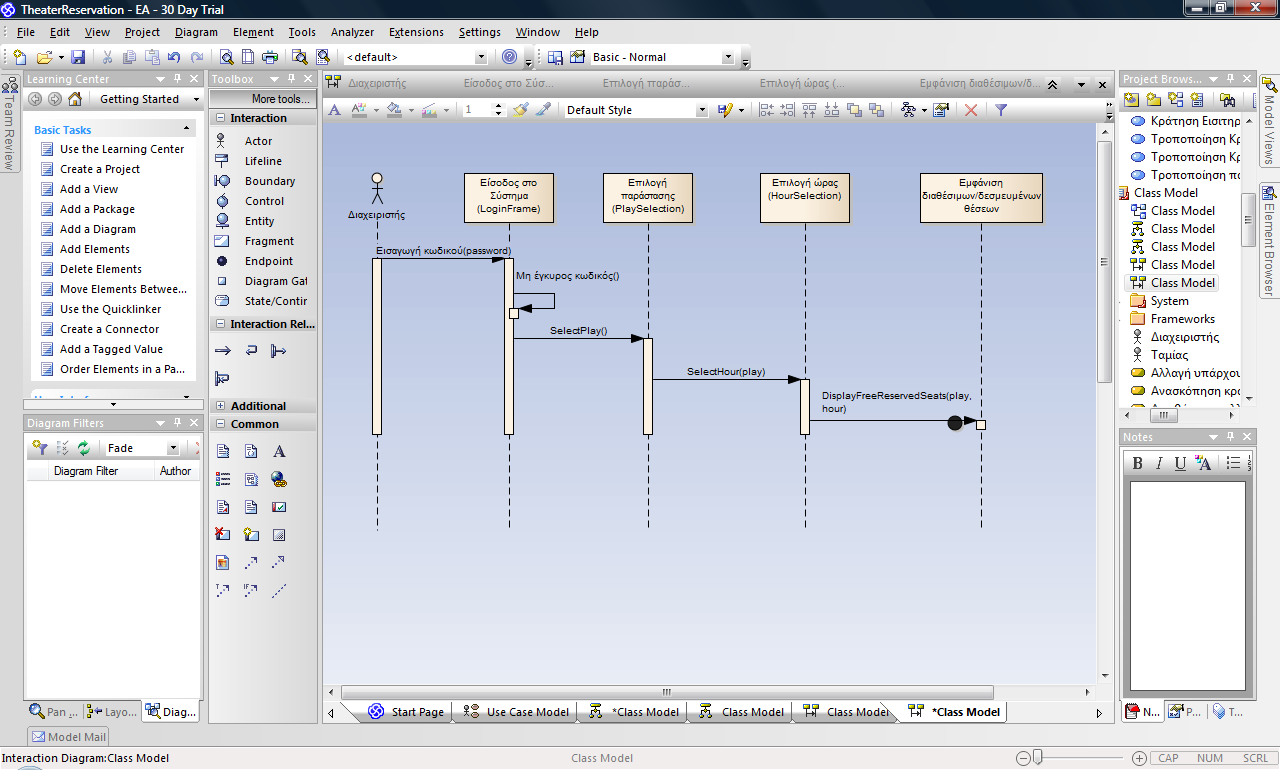


Εικόνα 2. UML Διάγραμμα κλάσεων

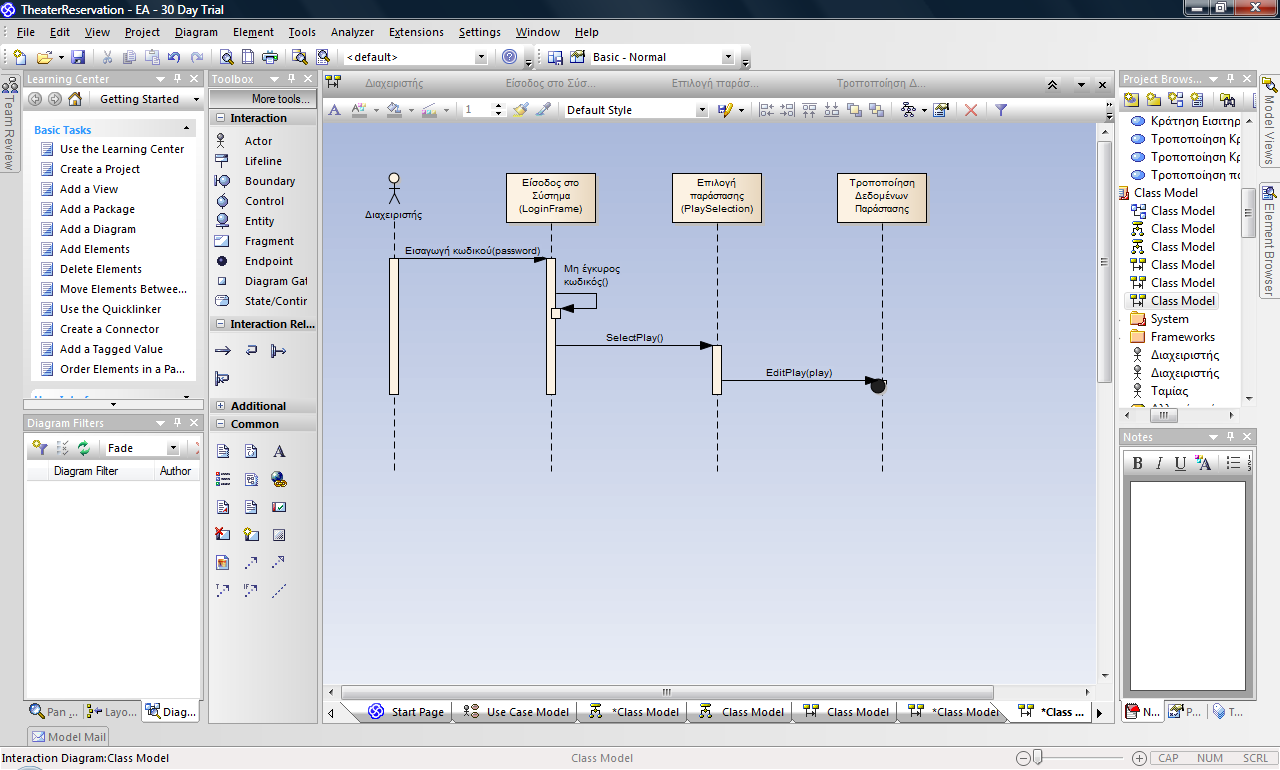
## UML Διαγράμματα Ακολουθίας



Εικόνα 3. UML Διάγραμμα Ακολουθίας για τον Ταμία

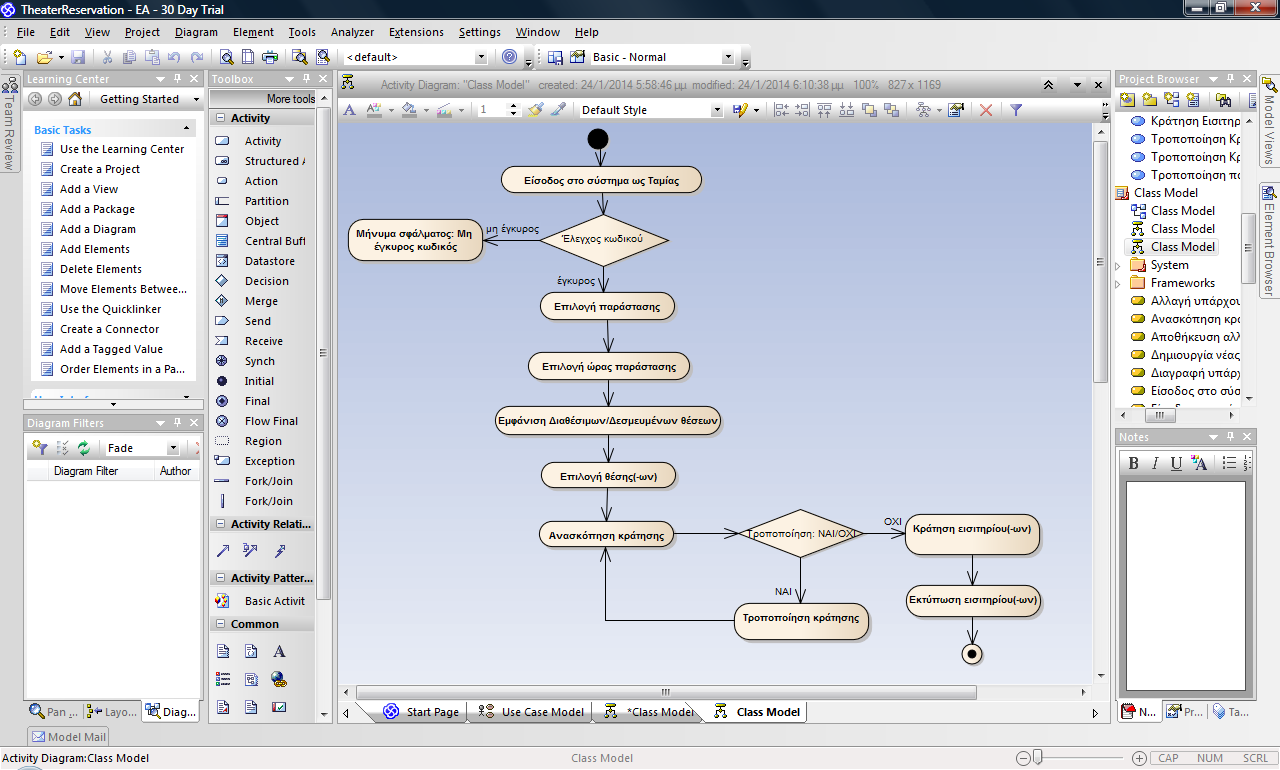


Εικόνα 4. UML Διάγραμμα Ακολουθίας για τον Διαχειριστή για την εμφάνιση των διαθέσιμων/δεσμευμένων θέσεων

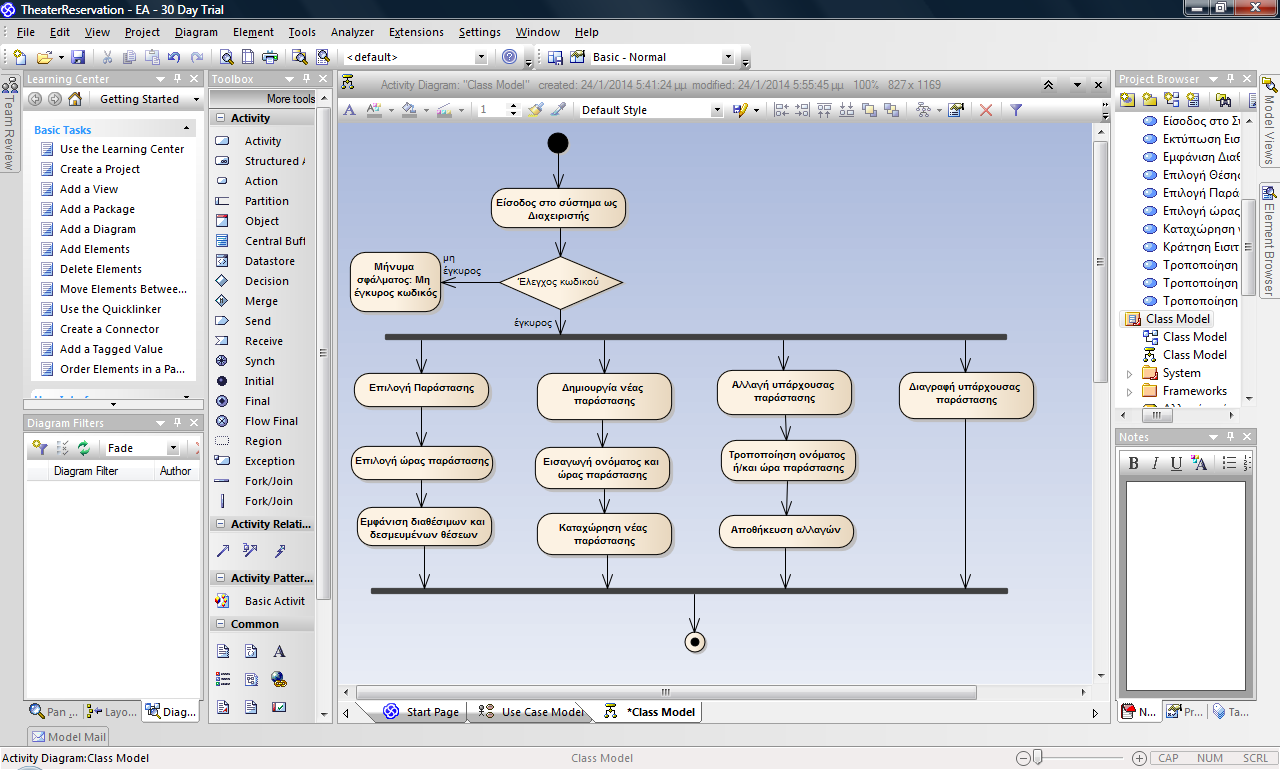


Εικόνα . UML Διάγραμμα ακολουθίας για τον Διαχειριστή για την τροποποίηση μιας παράστασης

## UML Διαγράμματα Δραστηριοτήτων



Εικόνα 6. UML Διάγραμμα Δραστηριοτήτων για τον Ταμία



Εικόνα 7. UML Διάγραμμα δραστηριοτήτων για τον Διαχειριστή του συστήματος

# Unit-testing

Το Unit-testing είναι μια μέθοδος όπου ξεχωριστές ενότητες (units) πηγαίου κώδικα ελέγχονται για να εξακριβωθεί ένα δουλεύουν έτσι όπως προβλέπεται. Στόχος είναι η δημιουργία unit-tests για συγκεκριμένες μεθόδους κλάσεων οι οποίες θεωρείται σημαντικό να ελεγχτούν για το αν τα αποτελέσματα τα οποία δίνουν είναι σύμφωνα με αυτά που προβλέπεται να επιστρέψουν.

***Unit-test για την κλάση LoginFrame***

package ticketsreservation;

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.Test;

import static org.junit.Assert.\*;

public class LoginFrameTest {

public LoginFrameTest() {

}

@BeforeClass

public static void setUpClass() {

System.out.println("Starting...");

}

@AfterClass

public static void tearDownClass() {

System.out.println("Done!");

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class LoginFrame.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

LoginFrame.main(args);

// TODO review the generated test code and remove the default call to fail.

fail("The test case is a prototype.");

}

}

***Unit-test για την κλάση PlaySelection***

package ticketsreservation;

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.Test;

import static org.junit.Assert.\*;

public class PlaySelectionTest {

public PlaySelectionTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class PlaySelection.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

PlaySelection.main(args);

// TODO review the generated test code and remove the default call to fail.

fail("The test case is a prototype.");

}

}

***Unit-test για την κλάση HourSelection***

package ticketsreservation;

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.Test;

import static org.junit.Assert.\*;

public class HourSelectionTest {

public HourSelectionTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class HourSelection.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

HourSelection.main(args);

// TODO review the generated test code and remove the default call to fail.

fail("The test case is a prototype.");

}

}

***Unit-test για την κλάση SeatSelection***

package ticketsreservation;

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.Test;

import static org.junit.Assert.\*;

public class SeatSelectionTest {

public SeatSelectionTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class SeatSelection.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

SeatSelection.main(args);

// TODO review the generated test code and remove the default call to fail.

fail("The test case is a prototype.");

}

}

***Unit-test για την κλάση AdminFrame***

package ticketsreservation;

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AdminFrameTest {

public AdminFrameTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class AdminFrame.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

AdminFrame.main(args);

// TODO review the generated test code and remove the default call to fail.

fail("The test case is a prototype.");

}

}

# Παράρτημα Α’ – Πηγαίος Κώδικας

***Κλάση LoginFrame***

package ticketsreservation;

public class LoginFrame extends javax.swing.JFrame {

/\*\*

\* Creates new form LoginFrame

\*/

public LoginFrame() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

btnGroup = new javax.swing.ButtonGroup();

jPanel1 = new javax.swing.JPanel();

rdbtnAdmin = new javax.swing.JRadioButton();

rdbtnCashier = new javax.swing.JRadioButton();

jLabel1 = new javax.swing.JLabel();

btnInsert = new javax.swing.JButton();

jLabel2 = new javax.swing.JLabel();

txtPassword = new javax.swing.JTextField();

btnGroup.add(rdbtnAdmin);

btnGroup.add(rdbtnCashier);

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Είσοδος στο σύστημα");

rdbtnAdmin.setSelected(true);

rdbtnAdmin.setText("Διαχειριστής");

rdbtnCashier.setText("Ταμίας");

jLabel1.setText("Είσοδος στο σύστημα ως:");

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(29, 29, 29)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(rdbtnCashier)

.addComponent(rdbtnAdmin))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addGroup(jPanel1Layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(0, 32, Short.MAX\_VALUE))

);

jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(13, 13, 13)

.addComponent(jLabel1)

.addGap(18, 18, 18)

.addComponent(rdbtnAdmin)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(rdbtnCashier)

.addContainerGap(22, Short.MAX\_VALUE))

);

btnInsert.setText("Είσοδος");

btnInsert.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnInsertActionPerformed(evt);

}

});

jLabel2.setText("Κωδικός:");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel2))

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(txtPassword, javax.swing.GroupLayout.PREFERRED\_SIZE, 116, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(110, Short.MAX\_VALUE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addComponent(btnInsert)

.addGap(30, 30, 30))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(txtPassword, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 26, Short.MAX\_VALUE)

.addComponent(btnInsert)

.addGap(23, 23, 23))

);

pack();

}// </editor-fold>

private void btnInsertActionPerformed(java.awt.event.ActionEvent evt) {

String text = txtPassword.getText();

String pwdCashier="cashier";

String pwdAdmin="admin";

if(rdbtnCashier.isSelected()){

if (text.equals(pwdCashier)) {

LoginFrame.this.setVisible(false);

PlaySelection playSelection= new PlaySelection();

playSelection.setVisible(true);

}else {

txtPassword.setText("Invalid Password!");

}

}else {

if(rdbtnAdmin.isSelected()){

if (text.equals(pwdAdmin)) {

LoginFrame.this.setVisible(false);

AdminFrame adminFrame= new AdminFrame();

adminFrame.setVisible(true);

}else {

txtPassword.setText("Invalid Password!");

}

}

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(LoginFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new LoginFrame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.ButtonGroup btnGroup;

private javax.swing.JButton btnInsert;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JPanel jPanel1;

private javax.swing.JRadioButton rdbtnAdmin;

private javax.swing.JRadioButton rdbtnCashier;

private javax.swing.JTextField txtPassword;

// End of variables declaration

}

***Κλάση AdminFrame***

package ticketsreservation;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class AdminFrame extends javax.swing.JFrame {

/\*\*

\* Creates new form AdminFrame

\*/

public AdminFrame() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

txtName = new javax.swing.JTextField();

jLabel2 = new javax.swing.JLabel();

txtHour = new javax.swing.JTextField();

btnRegister = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Εισαγωγή νέας παράστασης");

jLabel1.setText("Εισάγετε το όνομα της νέας παράστασης:");

jLabel2.setText("Εισάγετε την ώρα παρουσίασης της παράστασης:");

btnRegister.setText("Καταχώρηση");

btnRegister.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

btnRegisterActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(txtName)

.addComponent(jLabel2, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(txtHour))

.addComponent(btnRegister))

.addContainerGap(38, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(24, 24, 24)

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(txtName, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(txtHour, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(27, 27, 27)

.addComponent(btnRegister)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

} private void btnRegisterActionPerformed(java.awt.event.ActionEvent evt) {

Connection conn = null;

Statement stmt = null;

ResultSet rs = null;

try

{

Class.forName("com.mysql.jdbc.Driver").newInstance();

String connectionUrl = "jdbc:mysql://localhost:3306/theater";

String connectionUser = "root";

String connectionPassword = "root";

conn = DriverManager.getConnection(connectionUrl, connectionUser,

connectionPassword);

stmt = conn.createStatement();

String name= txtName.getText();

String hour= txtHour.getText();

String strSql1 = "INSERT INTO plays (name) values('"+name+"')";

stmt.executeUpdate(strSql1);

String strSql2 = "INSERT INTO time (play,time) values('"+name+"','"+hour+"')";

stmt.executeUpdate(strSql2);

stmt.close();

}

catch (Exception e)

{

e.printStackTrace();

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(AdminFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(AdminFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(AdminFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(AdminFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new AdminFrame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnRegister;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JTextField txtHour;

private javax.swing.JTextField txtName;

// End of variables declaration

}

***Κλάση PlaySelection***

package ticketsreservation;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.swing.DefaultListModel;

import javax.swing.JList;

public class PlaySelection extends javax.swing.JFrame {

public String selectedPlay = "Default";

//private DefaultListModel listModel;

/\*\*

\* Creates new form PlaySelection

\*/

public PlaySelection() {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

playsList = new javax.swing.JList();

jButton1 = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Επιλογή παράστασης");

Connection conn = null;

Statement stmt = null;

ResultSet rs = null;

DefaultListModel listModel = new DefaultListModel();

try

{

Class.forName("com.mysql.jdbc.Driver").newInstance();

String connectionUrl = "jdbc:mysql://localhost:3306/theater";

String connectionUser = "root";

String connectionPassword = "root";

conn = DriverManager.getConnection(connectionUrl, connectionUser,

connectionPassword);

stmt = conn.createStatement();

rs = stmt.executeQuery("SELECT \* FROM plays");

while (rs.next())

{

String name = rs.getString("name");

listModel.addElement(name);

}

//playsList = new JList(listModel);

}

catch (Exception e)

{

e.printStackTrace();

}

playsList.setModel(listModel);

jScrollPane1.setViewportView(playsList);

jButton1.setText("OK");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jLabel1.setText("Επιλέξτε την παράσταση για την οποία θέλετε να κάνετε κράτηση: ");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(0, 0, Short.MAX\_VALUE)

.addComponent(jButton1)

.addGap(145, 145, 145))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1))

.addGroup(layout.createSequentialGroup()

.addGap(45, 45, 45)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 238, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 14, Short.MAX\_VALUE)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 201, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(18, 18, 18)

.addComponent(jButton1)

.addGap(19, 19, 19))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

PlaySelection.this.setVisible(false);

selectedPlay = playsList.getSelectedValue().toString();

HourSelection hourSelection= new HourSelection(selectedPlay);

hourSelection.setVisible(true);

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(PlaySelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(PlaySelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(PlaySelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(PlaySelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new PlaySelection().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton1;

private javax.swing.JLabel jLabel1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JList playsList;

// End of variables declaration

}

***Κλάση HourSelection***

package ticketsreservation;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

import javax.swing.DefaultListModel;

public class HourSelection extends javax.swing.JFrame {

/\*\*

\* Creates new form HourSelection

\*/

public HourSelection(String play) {

initComponents(play);

}

@SuppressWarnings("unchecked")

private void initComponents(final String play) {

jScrollPane1 = new javax.swing.JScrollPane();

hourList = new javax.swing.JList();

jLabel1 = new javax.swing.JLabel();

btnOK = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Επιλογή ώρας παράστασης");

Connection conn = null;

Statement stmt = null;

ResultSet rs = null;

DefaultListModel listModel = new DefaultListModel();

try

{

Class.forName("com.mysql.jdbc.Driver").newInstance();

String connectionUrl = "jdbc:mysql://localhost:3306/theater";

String connectionUser = "root";

String connectionPassword = "root";

conn = DriverManager.getConnection(connectionUrl, connectionUser,

connectionPassword);

stmt = conn.createStatement();

rs = stmt.executeQuery("SELECT \* FROM time where play='"+play+"'");

while (rs.next())

{

String time = rs.getString("time");

listModel.addElement(time);

}

}

catch (Exception e)

{

e.printStackTrace();

}

hourList.setModel(listModel);

jScrollPane1.setViewportView(hourList);

jLabel1.setText("Επιλέξτε την ώρα της παράστασης:");

btnOK.setText("ΟΚ");

btnOK.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

HourSelection.this.setVisible(false);

String selectedHour = hourList.getSelectedValue().toString();

SeatSelection seatSelection= new SeatSelection(play,selectedHour);

seatSelection.setVisible(true);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(88, 88, 88)

.addComponent(btnOK))

.addGroup(layout.createSequentialGroup()

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 160, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jLabel1))))

.addContainerGap(28, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 104, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(32, 32, 32)

.addComponent(btnOK)

.addGap(24, 24, 24))

);

pack();

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(HourSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(HourSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(HourSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(HourSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new HourSelection("Default").setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton btnOK;

private javax.swing.JList hourList;

private javax.swing.JLabel jLabel1;

private javax.swing.JScrollPane jScrollPane1;

// End of variables declaration

}

***Κλάση SeatSelection***

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package ticketsreservation;

public class SeatSelection extends javax.swing.JFrame {

/\*\*

\* Creates new form SeatSelection

\*/

public SeatSelection(String selectedPlay, String selectedHour) {

initComponents();

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jScrollPane1 = new javax.swing.JScrollPane();

jList1 = new javax.swing.JList();

jScrollPane2 = new javax.swing.JScrollPane();

jList2 = new javax.swing.JList();

jScrollPane3 = new javax.swing.JScrollPane();

jList3 = new javax.swing.JList();

jScrollPane4 = new javax.swing.JScrollPane();

jList4 = new javax.swing.JList();

jScrollPane5 = new javax.swing.JScrollPane();

jList5 = new javax.swing.JList();

jScrollPane6 = new javax.swing.JScrollPane();

jList6 = new javax.swing.JList();

jScrollPane7 = new javax.swing.JScrollPane();

jList7 = new javax.swing.JList();

jScrollPane8 = new javax.swing.JScrollPane();

jList8 = new javax.swing.JList();

jScrollPane9 = new javax.swing.JScrollPane();

jList9 = new javax.swing.JList();

jScrollPane10 = new javax.swing.JScrollPane();

jList10 = new javax.swing.JList();

jScrollPane11 = new javax.swing.JScrollPane();

jList11 = new javax.swing.JList();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setTitle("Επιλογή θέσης");

jList1.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "1", "12", "23", "34"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane1.setViewportView(jList1);

jList2.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "2", "13", "24", "35"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane2.setViewportView(jList2);

jList3.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "3", "14", "25", "36"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane3.setViewportView(jList3);

jList4.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "4", "15", "26", "37"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane4.setViewportView(jList4);

jList5.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "5", "16", "27", "38"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane5.setViewportView(jList5);

jList6.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "7", "18", "29", "40"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane6.setViewportView(jList6);

jList7.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "6", "17", "28", "39"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane7.setViewportView(jList7);

jList8.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "8", "19", "30", "41"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane8.setViewportView(jList8);

jList9.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "11", "22", "33", "44"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane9.setViewportView(jList9);

jList10.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "10", "21", "32", "43"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane10.setViewportView(jList10);

jList11.setModel(new javax.swing.AbstractListModel() {

String[] strings = { "9", "20", "31", "42"};

public int getSize() { return strings.length; }

public Object getElementAt(int i) { return strings[i]; }

});

jScrollPane11.setViewportView(jList11);

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N

jLabel1.setText("ΑΡΙΣΤΕΡΑ");

jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N

jLabel2.setText("ΚΕΝΤΡΟ");

jLabel3.setFont(new java.awt.Font("Tahoma", 1, 11)); // NOI18N

jLabel3.setText("ΔΕΞΙΑ");

jLabel4.setText("Επιλέξτε την θέση ή τις θέσεις για κράτηση:");

jButton2.setText("Κράτηση");

jButton3.setText("Εκτύπωση");

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel1)

.addGap(167, 167, 167)

.addComponent(jLabel2))

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(55, 55, 55)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButton2)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane4, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane5, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane7, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane6, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane8, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addComponent(jButton3))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 43, Short.MAX\_VALUE)

.addComponent(jScrollPane11, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jScrollPane10, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane9, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addComponent(jLabel3))

.addGap(23, 23, 23))

.addGroup(layout.createSequentialGroup()

.addComponent(jLabel4)

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(19, 19, 19)

.addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jLabel1)

.addComponent(jLabel2)

.addComponent(jLabel3))

.addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane9, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane10, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane11, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane8, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane7, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane6, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane5, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane4, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane3, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 95, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(38, 38, 38)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)

.addComponent(jButton2)

.addComponent(jButton3))

.addGap(20, 20, 20))

);

pack();

}// </editor-fold>

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(SeatSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(SeatSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(SeatSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(SeatSelection.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new SeatSelection("Default", "Default").setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JList jList1;

private javax.swing.JList jList10;

private javax.swing.JList jList11;

private javax.swing.JList jList2;

private javax.swing.JList jList3;

private javax.swing.JList jList4;

private javax.swing.JList jList5;

private javax.swing.JList jList6;

private javax.swing.JList jList7;

private javax.swing.JList jList8;

private javax.swing.JList jList9;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JScrollPane jScrollPane10;

private javax.swing.JScrollPane jScrollPane11;

private javax.swing.JScrollPane jScrollPane2;

private javax.swing.JScrollPane jScrollPane3;

private javax.swing.JScrollPane jScrollPane4;

private javax.swing.JScrollPane jScrollPane5;

private javax.swing.JScrollPane jScrollPane6;

private javax.swing.JScrollPane jScrollPane7;

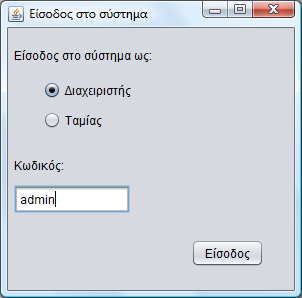
private javax.swing.JScrollPane jScrollPane8;

private javax.swing.JScrollPane jScrollPane9;

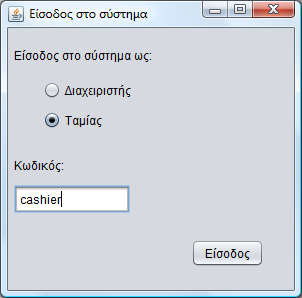
// End of variables declaration

}

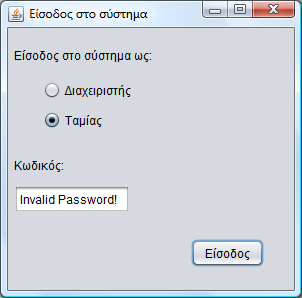
# Παράρτημα Β’ – Screenshots του Συστήματος



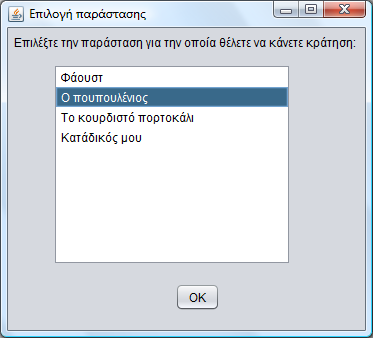
Εικόνα . Είσοδος στο Σύστημα ως Διαχειριστής



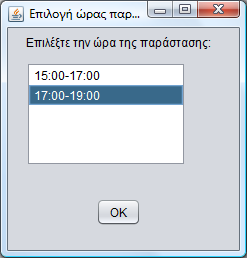
Εικόνα . Είσοδος στο Σύστημα ως Ταμίας



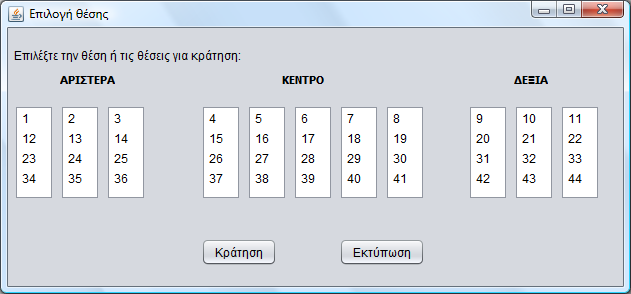
Εικόνα . Εισαγωγή λάθους κωδικού



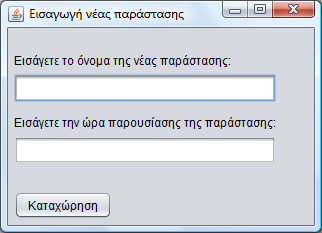
Εικόνα . Επιλογή παράστασης



Εικόνα . Επιλογή ώρας παράστασης



Εικόνα . Επιλογή θέσης



Εικόνα . Εισαγωγή νέας παράστασης