**ΤΕΙ ΠΕΙΡΑΙΑ**

# ΗΛΕΚΤΡΟΝΙΚΑ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

ΕΡΓΑΣΤΗΡΙΟ ΜΗΧΑΝΙΚΗΣ ΛΟΓΙΣΜΙΚΟΥ



**ΕΡΓΟ:** **Σύστημα Κρατήσεων Εισιτηρίων ΚΤΕΛ**

**ΣΠΟΥΔΑΣΤΕΣ:**

**ΜΕΤΣΗ ΣΙΛΒΙΤΑ 38364**

**ΜΑΡΙΝΑ ΑΡΝΤΙΤ 36165**

**ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ 38469**

**ΤΜΗΜΑ:**

**ΔΕΥΤΕΡΑ 20.00-22.00**

# 

*Χειμερινό Εξάμηνο - Φεβρουάριος 2013*

***Περιεχόμενα***

1.0 Εισαγωγή 4

1.1 Σκοπός του κειμένου 4

1.2 Προορισμένο ακροατήριο και προτάσεις ανάγνωσης 4

1.3 Στόχος και εύρος του προϊόντος 5

2.0 Γενική Περιγραφή 5

2.1 Προέλευση και προοπτική του προϊόντος 5

2.2 Γενική περιγραφή των λειτουργιών του προϊόντος.................................................................................5

2.3 Κατηγορίες χρηστών. 6

2.4 Περιορισμοί στη σχεδίαση και την υλοποίηση .6

2.5 Προϋποθέσεις – Εξαρτήσεις 7

3.0 Απαιτήσεις για τις εξωτερικές διεπαφές 7

3.1 Διεπαφή χρήστη. 7

3.2 Διεπαφές υλικού 8

3.3 Διεπαφές λογισμικού 8

4.0 Λειτουργίες συστήματος. 8

4.1 Κράτηση Εισιτηριων. 9

Περιγραφή και προτεραιότητα.

Ακολουθίες ερεθίσματος / απόκρισης

5.0 Μη λειτουργικές απαιτήσεις 10

5.1 Επιδόσεις. 10

5.2 Φυσική ασφάλεια 11

5.4 Ποιότητα λογισμικού 11

6.0 Kατάλογος περιπτώσεις χρήσης 13

7.0 Καρτες CRC-Διαγραμματα 22

7.1 Απλοποιήμενες Καρτες CRC 23

7.2 Διαγραμματα UML 25

8.0 Κωδικας Κλασεων 28

8.1 Kωδικας κλασηςClient 31

8.2 Kωδικας κλασης Home 32

8.3 Kωδικας κλασης GUI 39

9.0 Interface 47

9.1 Debugging ……………………………………………………………………………………………………………………………….……..50

10 Γενικά 50

**Έντυπο Απαιτήσεων**

**για το**

**Σύστημα κρατήσεων εισιτηρίων**

Από TURBO IT Solutions

Version 1.1 approved

Δημιουργήθηκε από ΟΜΑΔΑ 59(ΜΕΤΣΗ Σ., ΜΑΡΙΝΑ Α., ΚΩΝΣΤΑΝΤΑΡΑ Α.)

TURBO IT Solutions,

ΦΕΒΡΟΥΑΡΙΟΣ 2013

*Αναθεωρήσεις*

|  |  |  |  |
| --- | --- | --- | --- |
| **Συγγραφέας** | **Ημερομηνία** | **Αίτια αναθεωρήσεων** | **Έκδοση** |
| ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ | 03/11/2012 | ΠΡΟΧΕΙΡΟ | Draft 0.1 |
| ΜΕΤΣΗ ΣΙΛΒΙΤΑ | 5/11/2012 | Αναθεώρηση Προχείρου, Συμπλήρωση Κεφαλαίου 2.2, Ανάπτυξη Λειτουργιών | Draft 1.0 |
| ΜΑΡΙΝΑ ΑΡΝΤΙΤ | 23/12/2012 | Συμπλήρωση 4ου Κεφαλαίου | Draft 1.1 |

# 1.0 Εισαγωγή

## 1.1 Σκοπός του κειμένου

Ο σκοπός αυτού του εγγράφου είναι ο προσδιορισμός όλων των απαιτήσεων του λογισμικού συστήματος κρατήσεων εισιτηρίων έκδοσης 1.1( **ticket booking system**), το οποίο είναι ένα πρόγραμμα κράτησης εισιτηρίων το οποίο επιτρέπει την αυτοματοποίηση της διαδικασίας κρατήσεων μέσα από τον υπάρχοντα χώρο, μειώνοντας το χρόνο και το κόστος διεκπεραίωσης τους. Η **Εφαρμογή ηλεκτρονικών κρατήσεων** απευθύνεται σε επιχειρήσεις οι οποίες έχουν σαν στόχο την αναβάθμιση των υφισταμένων ή τη δημιουργία νέων real time υπηρεσιών με το πιο απλό, γρήγορο, ασφαλές και οικονομικό τρόπο. Το πρόγραμμα αυτό χρησιμοποιώντας ως κύριο χαρακτηριστικό την αλληλεπίδραση με τον χρήστη θα προσφέρει μια απλή διαδικασία κράτησης με τρόπο κατανοητό, σαφή, πλήρη και συνεπή. Αυτό το έγγραφο ακολουθεί το πρότυπο ΙΕΕΕ για έγγραφα προδιαγραφής απαιτήσεων με κάποιες παραλλαγές .

## 1.2 Προορισμένο ακροατήριο και προτάσεις ανάγνωσης

Αυτό το έγγραφο απευθύνεται τόσο στους μηχανικούς λογισμικού όσο και στους διευθυντές προγράμματος, το εμπορικό προσωπικό, τους συγγραφείς τεκμηρίωσης και στον πελάτη-χρήστη του προγράμματος. Αποτελεί το μοναδικό έγγραφο που περιγράφει τις απαιτήσεις του λογισμικού κράτησης εισιτηρίων. Μπορεί να χρησιμοποιηθεί από τους προγραμματιστές που θα αναπτύξουν το λογισμικό και θα χρησιμεύσει για την επικύρωση του λογισμικού που θα παραδοθεί στον πελάτη-χρήστη. Κάθε αλλαγή στις απαιτήσεις μελλοντικά είναι απαραίτητο να γίνει από μία τυπική διαδικασία αλλαγής και αποδοχής αυτού του εγγράφου. Η ακολουθία για την ανάγνωση του εγγράφου είναι ίδια για όλους τους χρήστες πλην του εμπορικού προσωπικού και των απλών χρηστών που θα πρέπει να δώσουν ιδιαίτερη έμφαση στο εισαγωγικό κομμάτι του εγγράφου.

## 

## 1.3 Στόχος και εύρος του προϊόντος

Ο στόχος του προγράμματος κράτησης εισιτηρίων είναι η αυτοματοποιημένη διαδικασία κράτησης και κοστολόγησης διαδρομών μιας μονάδας λεωφορείων , με τέτοιο τρόπο που διασφαλίζετε η απλότητα ,η λειτουργικότητα και η ταχύτητα της προαναφέρουσας διαδικασίας. Στόχος μας είναι να δώσουμε στους χρήστες την δυνατότητα της κράτησης εισιτηρίων απο τον χώρο αναχώρησης οποιαδήποτε χρονική στιγμή.

# 2.0 Γενική Περιγραφή

## 2.1 Προέλευση και προοπτική του προϊόντος

To πρόγραμμα κράτησης εισιτηρίων αποτελεί ένα δυναμικά ανεξάρτητο και αυτόνομο τμήμα μιας σείρας εφαρμογών διαχείρησης μονάδων από την νέα πλατφόρμα της TURBO. Οι δυνατότητες της πλήρους γκάμας των προγραμμάτων μας επεκτείνονται σε εμπορικά προγράμματα  πλήρους διαχείρησης υπηρεσίων.

## 2.2 Γενική περιγραφή των λειτουργιών του προϊόντος

**Οι σημαντικότερες λειτουργίες που θα επιτελεί το αυτοματοποιημένο σύστημα είναι οι :**

* Κράτηση εισιτηρίων
* Διόρθωση / Ακύρωση κράτησης
* Επιβεβαίωση κράτησης
* Υπολογισμός συνολικού κόστους διαδρομής
* Εκτύπωση Εισιτηρίου

## 2.3 Κατηγορίες χρηστών

Όσον αφορά το προσωπικό της επιχείρησης θα δοθούν κωδικοί πρόσβασης, αλλά κάθε κωδικός θα παρέχει περιορισμένη πρόσβαση. Συγκεκριμένα:

* Οι υπάλληλοι στο ταμείο θα διαχειρίζονται μόνο απλές λειτουργίες όπως εισαγωγή στοιχείων , επιβεβαίωση των κρατήσεων και εκτύπωση εισιτηρίου .
* Ο υπεύθυνος θα μπορεί να διαχειρίζεται τις τιμές των διαδρομών και τις προσφορές .
* Ο διευθυντής της επιχείρησης θα έχει πλήρη πρόσβαση του λογισμικού εκτός από τα τεχνικά χαρακτηριστικά του.
* Ο υπεύθυνος διαχείρισης (admin) του λογισμικού θα έχει πρόσβαση μόνο στα τεχνικά χαρακτηριστικά του λογισμικού.

## 2.4 Περιορισμοί στη σχεδίαση και την υλοποίηση

Το πρόγραμμα κράτησης εισιτηρίων θα εκτελείται **ΤΟΠΙΚΑ** σε υπολογίστες με επεξεργαστή της σειράς Pentium ΙΙΙ και άνω, με συχνότητα λειτουργίας 600 ΜΗΖ ή περισσότερο, και λειτουργικό σύστημα Microsoft Windows 98, ΝΤ, 2000, ΧΡ, Vista, Windows 7.H ελάχιστη μνήμη RAM που θα απαιτείται θα είναι 128 MB και ο συνολικός χώρος που θα καταλαμβάνει το πρόγραμμα θα είναι τα 40 MB. Αναφορικά με τους περιορισμούς σχεδιασμού και υλοποίησης, η άδεια του προγράμματος είναι αυστηρώς ιδιωτική, δηλαδή η χρήση του τοπικού προγράμματος διαχείρησης θα επιτρέπεται μονο στους υπολογιστές της επιχείρησης και απαγορεύεται η αντιγραφή και η μεταπώληση του. Όποιος ασχολείται ή πρόκειται να ασχοληθεί με την διαχείρηση του συγκεκριμένου έργου, καθώς και με τη χρήση του θα πρέπει να αποδέχεται τους όρους της άδειας εφόσον αποτελεί έναν από τους σημαντικότερους περιορισμούς.

## 

## 2.5 Προϋποθέσεις – Εξαρτήσεις

Το πρόγραμμα, το όποιο θα χρησιμοποιείται τοπικά στους υπολογιστές και θα είναι υπεύθυνο για την διαχείρηση και την τροποποίηση του κυρίου προγράμματος κρατήσεων εισιτηρίων, δεν θα έχει κάποιες προυποθέσεις ούτε εξαρτήσεις απο πλεύρας λογισμικού. Η μόνη απαίτηση είναι η ύπαρξη σύνδεσης με το δίκτυο αν και εφόσον είναι παραπάνω από ένας σταθμός, ωστε το πρόγραμμα να συνδεθεί επιτυχώς στον εξυπηρετητή που θα φιλοξενεί την Βάση Δεδομένων των διαδρομών μας .

# 3.0 Απαιτήσεις για τις εξωτερικές διεπαφές

## 3.1 Διεπαφή χρήστη

**Η διεπαφή του χρήστη θα περιλαμβάνει:**

Α) Έ**να παράθυρο με την απεικόνιση του νομού**: στο οποίο θα φαίνονται οι νομοί που θα μπορούν να επιλέξουν οι χρήστες για να κάνουν κράτηση ενός εισιτηρίου .

Β) **Κουμπιά επιλογών**: τα οποία θα χρησιμοποιεί ο χρήστης για να εκτελέσει διάφορες λειτουργίες,οπως επιβεβαίωση κράτησης,διόρθωση κράτησης,ακύρωση κράτησης.

Γ) Έ**να κουμπί εμφάνισης πληροφοριών**: με αυτό ο χρήστης θα επιλέγει να εμφανίσει τη φόρμα πληροφοριών για τις διαδρομές . Αυτή θα περιέχει πληροφορίες σχετικά με την διαδρομή ,τις ώρες ,την τιμή , την προσφορά της ημέρας κ.α

Δ) **Ένα κουμπί ελέγχου διαθεσιμότητας**: με αυτό το κουμπί ο χρήστης θα ελέγχει την διαθεσιμότητα των λεωφορείων για κάποια συγκεκριμένη ημερομηνία και διαδρομη .

Ε) Έ**να κουμπί εξόδου / τερματισμού προγράμματος**: με αυτό ο χρήστης θα τερματίζει το πρόγραμμα.

## 3.2 Διεπαφές υλικού

Το λογισμικό θα έχει κάποιες απαιτήσεις από την πλευρά του υπολογιστικού συστήματος στο οποίο θα τρέχει. Το λογισμικό εγγυάται ότι θα ανταποκρίνεται με τον καλύτερο δυνατό τρόπο σε περίπτωση που το σύστημα αποτυγχάνει.Το πρόγραμμα θα λειτουργεί σε όλους τους ηλεκτρονικούς υπολογιστές με ελάχιστες απαιτήσεις του επεξεργαστή. Θα απαιτεί την ύπαρξη ενός πληκτρολογίου με πρότυπο τα 104 κουμπιά, ένα ποντίκι με τουλάχιστον 2 κουμπιά (δεξί και αριστερό κλικ), μια οθόνη, μια κάρτα γραφικών, ενώ απαραίτητα είναι και τα ηχεία ώστε να υπάρχουν οι δυνατότητες των ειδοποιήσεων από εισαγωγή εξωτερικού τερματικού.

## 3.3 Διεπαφές λογισμικού

Η συγκεκριμένη διεπαφή εξαρτάται άμεσα από την διεπαφή του υλικού όσο και από την πλατφόρμα λογισμικού. Το συγκεκριμένο λογισμικό θα λειτουργεί ικανοποιητικά σε λειτουργικά συστήματα windows 98 και άνω. Το πρόγραμμα στο παρασκήνιο θα συλλέγει πληροφορίες για την διαθεσιμότητα των θέσεων ,τις τιμές κ.α από έναν εξυπηρετητή με βάση δεδομένων (SQL Database). Όταν ο χρήστης θα ολοκληρώνει μια κράτηση η βάση δεδομένων θα ενημερώνετε αυτόματα απο το σύστημα.

# 4.0 Λειτουργίες συστήματος

Το δικτυακό πρόγραμμα θα είναι προσβάσιμο από τους χρήστες μέσω τερματικού αφής της, κάνοντας «κλικ» στο κουμπί υπερσυνδέσμου **< KΡΑΤΗΣΗ>.** Πατώντας το κουμπί αυτό οι χρήστες θα περιηγηθούν στη κύρια σελίδα του προγράμματος, η οποία θα περιλαμβάνει ένα πλαίσιο δρομολογιών για τις εναλλαγές των επιθυμητών ημερομηνιών . Επίσης θα περιέχει την επιλογή του συνόλου των εισιτηρίων καθώς και το τύπο του κάθε εισιτηρίου (πχ φοιτητικό,απλό,προσφορά) και επίσης ένα κουμπί ελέγχου διαθεσιμότητας θέσεων. Eπίσης, θα εμπεριέχονται κουμπιά για επιβεβαίωση,αλλαγή, ακύρωση της κράτησης.

* Εύκολη διαχείριση-Αναζήτηση Κρατήσεων
* Ακύρωση ή διόρθωση κράτησης
* Εύκολη διαχείριση Διαθεσιμότητας ανά Διαδρομή
* Αυτόματη επιβεβαίωση
* Μέγιστος και ελάχιστος αριθμός κρατήσεων

## 4.1 Κράτηση Εισιτηρίων

### Περιγραφή και προτεραιότητα

Ο πελάτης εισέρχεται στο σύστημα, επιλέγει την ώρα για την οποία επιθυμεί να πραγματοποιήσει την διαδρομή , επιλέγει τον τύπο του εισιτηρίου , δηλώνει το πλήθος των ατόμων, αλλά και την διαδρομή την οποία θα πραγματοποιήσει , δηλώνει τρόπο πληρωμής και προσωπικά στοιχεία σχετικά με αυτόν (για παράδειγμα αριθμό πιστωτικής) και επιβεβαιώνει την κράτηση. Όταν επιβεβαιώσει την κράτηση, του αποστέλλεται ένα μήνυμα για συγκράτηση ορισμένων στοιχείων, όπως η ημερομηνία κράτησης και ο τύπος του εισιτηρίου που επέλεξε, και προχωρά η κατάσταση για την πληρωμή της κράτησης σύμφωνα με τον τρόπο που επέλεξε. Η πληρωμή επιτυγχάνεται με το πέρας ενός λεπτού από την στιγμή που έχει επιλέξει για την κράτηση του εισιτηρίου και την επιβεβαίωση της πληρωμής . Προτεραιότητα: Υψηλή - 1.

### Ακολουθίες ερεθίσματος / απόκρισης

Ερέθισμα: Ο πελάτης επιλέγει την ημέρα που θα πραγματοποιήσει την διαδρομή .

Απόκριση: Το σύστημα ελέγχει αν υπάρχει έστω μια κενή θέση την ημέρα εκείνη .

Ερέθισμα: Ο πελάτης επιλέγει πλήθος ατόμων και θέσεων για τα οποία επιθυμεί να κρατήσει εισιτηρια .

Απόκριση: Ενημερώνεται το μενού των θέσεων και εμφανίζονται δυνατές επιλογές στον πελάτη μόνο για όσα ανταποκρίνονται στην αναλογία άτομα/εισιτήρια (άτομα ανά εισιτήρια ).

Ερέθισμα: Ο πελάτης επιλέγει από το μενού ένα από τα διαθέσιμα δρομολόγια.

Απόκριση: Το σύστημα υπολογίζει το κόστος της διαδρομής σύμφωνα με τις προσφορές και την επιλογή του εισιτηρίου και εμφανίζει το συνολικό κόστος.

Ερέθισμα: Ο πελάτης ολοκληρώνει την διαδικασία επιλέγοντας να προχωρήσει στην κράτηση των εισιτηρίων .

Απόκριση: Ανοίγει μία νέα σελίδα που ζητά προσωπικά στοιχεία του πελάτη, όπως Ονοματεπώνυμο, θέση , διαδρομή.

Ερέθισμα: Ο πελάτης συμπληρώνει τα στοιχεία του και τον τρόπο πληρωμής όπως και τα προσωπικά του σχετικά στοιχεία (όπως ο αριθμός της πιστωτικής του κάρτας) και ολοκληρώνει την διαδικασία πατώντας το αντίστοιχο κουμπί.

Απόκριση: Το σύστημα του εμφανίζει σε νέα σελίδα τα στοιχεία που έχει επιλέξει όπως και τον τύπο πληρωμής και κάθε σχετικό στοιχείο συνολικά, προς επιβεβαίωσή της κράτησης.

Ερέθισμα: Ο πελάτης ελέγχει τα στοιχεία της κράτησης και επιβεβαιώνει την κράτηση.

Απόκριση: Αποστέλλεται μήνυμα εφόσον επιβεβαίωσε την κράτηση . Επίσης, μετά την επιβεβαίωση της κράτησής του, εμφανίζεται στον πελάτη τρόπος απόκτησης του εισιτηρίου για την διευκόλυνσή του. Επιπλέον, ενημερώνεται η βάση δεδομένων σχετικά με της διαθέσιμες θέσεις και ενημερώνεται το σύστημα .

Ερέθισμα: Ο πελάτης βλέπει τα συνολικά στοιχεία και επιλέγει αντί για επιβεβαίωση, διόρθωση των στοιχείων της κράτησης.

Απόκριση: Το σύστημα επαναφέρεται στην αρχική σελίδα κράτησης εισιτηρίων με επιλεγμένα τα στοιχεία που είχε συμπληρώσει προηγουμένως και επαναλαμβάνονται τα προηγούμενα βήματα.

5.0 Μη λειτουργικές απαιτήσεις

## 5.1 Επιδόσεις

Το λογισμικό δεν θα δεσμεύει μεγάλη ταχύτητα του λειτουργικού συστήματος καθώς οι απαιτήσεις του δεν είναι ιδιαίτερα υψηλές.

Το λογισμικό δεν θα πρέπει να καταλαμβάνει μεγάλο μέρος του σκληρού δίσκου έτσι ώστε να μην επιβαρύνει τις συνολικές επιδόσεις του συστήματος.

Backup: Η ενέργεια αυτή θα έχει ως αποτέλεσμα μια ελάχιστη καθυστέρηση (της τάξεως 0.35%-0.45%) στην απόκριση του λογισμικού και θα απαιτεί περισσότερο χώρο στο δίσκο. Ένα ολικό Backup θα έχει αύξηση της καθυστέρησης κατά 1.50%.

## 5.2 Φυσική ασφάλεια

Το λογισμικό διαθέτει backup σύστημα αρχείων το οποίο ενεργοποιείται και χειροκίνητα αλλά και αυτόματα. Με αυτόν τον τρόπο η απώλεια δεδομένων από φυσική ασφάλεια είναι σχεδόν αδύνατη. Επιπλέον το πρόγραμμα σε αρχεία κειμένου θα κρατάει αντίγραφα όλων των κρατήσεων μιας μη εκτελούμενης διαδρομής σε περίπτωση λανθασμένης διαγραφής τους. Φυσικά απαιτείται να υπάρχει επαρκής χώρος στο δίσκο για τις παραπάνω διεργασίες.

## 5.3 Ποιότητα λογισμικού

* Ευχρηστία

Το λογισμικό θα διαθέτει ένα φιλικό προς τον χρήστη περιβάλλον εργασίας καθώς για να το χειριστεί δεν χρειάζονται εξειδικευμένες γνώσεις πληροφορικής. Το interface του λογισμικού θα περιλαμβάνει παράθυρα, εικονίδια και μενού. Για να εξοικειωθεί ο χρήστης με το λογισμικό θα χρειάζεται ελάχιστος χρόνος.

* Δυνατότητα Συντήρησης

Το λογισμικό θα έχει δυνατότητα συντήρησης ώστε στο μέλλον να ικανοποιήσει και άλλες ανάγκες της εταιρίας.

* Προσαρμοστικότητα

Το λογισμικό θα έχει δυνατότητα να ανταποκρίνεται σε αλλαγές των απαιτήσεων του περιβάλλοντος (δυναμική αναβάθμιση του χωρίς να σταματήσει να λειτουργεί), να βοηθά τη γρήγορη ανάπτυξη νέου λογισμικού με βάση το παλιό και να διορθώνει σφάλματα που εμφανίζονται κατά τη διάρκεια λειτουργίας του.

* Αξιοπιστία
* Διαθεσιμότητα: Το σύστημα θα είναι διαθέσιμο για την λειτουργία του λογισμικού όταν το απαιτεί ο τελικός χρήστης.
* Συχνότητα αποτυχιών: Το σύστημα θα πρέπει να εγγυάται τις κατά το δυνατόν λιγότερες αποτυχίες του, όσο το λογισμικό βρίσκεται σε εξέλιξη από το χρήστη. Επομένως θα εξασφαλίζει στο χρήστη την ασφαλή λειτουργία του λογισμικού και την απαραίτητη προϋπόθεση ότι δεν θα χάνονται οι ενέργειες του χρήστη έπειτα από μια απρόσμενα αποτυχία.

Περιπτώσεις Χρήσης

για το

Σύστημα κρατήσεων εισιτηρίων

Από TURBO IT Solutions

Έκδοση 1.1

Αριθμός Ομάδας 59(ΜΕΤΣΗ Σ., ΜΑΡΙΝΑ Α., ΚΩΝΣΤΑΝΤΑΡΑ Α.)

Αναθεωρήσεις

|  |  |  |  |
| --- | --- | --- | --- |
| **Ονοματεπώνυμο** | **Ημερομηνία** | **Αιτία Αναθεώρησης** | **Έκδοση** |
| ΜΑΡΙΝΑ ΑΡΝΤΙΤ | 10/12/2012 | Draft | 0.1 |
| ΜΕΤΣΗ ΣΙΛΒΙΤΑ | 14/12/2012 | Εισαγωγή σχολίων | 1.0 |
| ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ | 15/12/2012 | Διορθώσεις | 1.1 |

**6.0 Κατάλογος Περιπτώσεων Χρήσης**

|  |  |
| --- | --- |
| **Εμπλεκόμενοι Ρόλοι** | **Περιπτώσεις Χρήσης** |
| Πελάτης | **UC-1.** Κράτηση εισιτηρίων  **UC-2.** Εισαγωγή πλήθους ατόμων  **UC-3.** Επιλογή διαδρομών .  **UC-6.** Εισαγωγή στοιχείων για κράτηση |
| Σύστημα Kρατήσεων Εισιτηρίων ΚΤΕΛ | **UC-4.** Εμφάνιση διαθέσιμων θέσεων  **UC-5.** Εμφάνιση λεπτομερειών διαδρομών  **UC-7.** Έλεγχος στοιχείων και κράτηση εισιτηρίων  **UC-8.** Είσοδος στο σύστημα. |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-1** | | |
| Ονομασία: | Κράτηση Θέσεων | | |
| Δημιουργήθηκε από: | ΜΑΡΙΝΑ ΑΡΝΤΙΤ | Τελευταία ενημέρωση από: | ΜΑΡΙΝΑ ΑΡΝΤΙΤ |
| Ημερομηνία Συγγραφής: | 10/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 22/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Πελάτης | | |
| Περιγραφή: | Ο Πελάτης μπορεί να εισέλθει στο σύστημα κρατήσεων εκτελώντας την εφαρμογή. | | |
| Γεγονός Εκκίνησης: | Ο Πελάτης εκτελεί την εφαρμογή Ticket | | |
| Προϋποθέσεις: | Η εφαρμογή να είναι εκτελέσιμη. | | |
| Τελική Κατάσταση: | 1. Ο πελάτης βλέπει το μενού επιλογών ατόμων για να κάνει κράτηση. 2. Ο πελάτης έχει επιλογή να δει της διαθέσιμες θέσεις ανά διαδρομή την οποία διάλεξε . 3. Ο πελάτης μπορεί να δει λεπτομέρειες για κάθε διαδρομή. 4. Ο πελάτης μπορεί να επιλέξει κράτηση του εισιτηρίου εισάγοντας ονοματεπώνυμο και θέση ανάλογα με την ώρα αναχώρησης από το σταθμό. | | |
| Φυσιολογική Ροή: | 1. UC - 8 (Είσοδος στο σύστημα) 2. UC - 2 (Εισαγωγή πλήθους ατόμων) 3. UC - 3 (Επιλογή διαδρομής ) 4. UC - 4 (Εμφάνιση διαθέσιμων θέσεων ) 5. UC - 5 (Εμφάνιση λεπτομερειών διαδρομής ) 6. UC - 6 (Εισαγωγή στοιχείων για κράτηση) 7. UC - 7 (Έλεγχος στοιχείων και κράτηση θέσεων ) | | |
| Εναλλακτική Ροή: | *Εμφάνιση στοιχείων όλων των διαθέσιμων διαδρομών :*   1. Είσοδος στο σύστημα 2. UC - 3 (Επιλογή διαδρομής ) 3. UC - 4 (Εμφάνιση διαθέσιμων θέσεων ) 4. UC - 5 (Εμφάνιση λεπτομερειών διαδρομής ) | | |
| Εξαιρέσεις: | Εξαιρούνται οι τυχόν μέρες συντήρησης της ιστοσελίδας. | | |
| Ενσωματώνει: | UC-2,UC-3, UC-4, UC-5, UC-6,UC-7 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 30 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-2** | | |
| Ονομασία: | Εισαγωγή πλήθους ατόμων | | |
| Δημιουργήθηκε από: | ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ | Τελευταία ενημέρωση από: | ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ |
| Ημερομηνία Συγγραφής: | 13/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 22/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Πελάτης | | |
| Περιγραφή: | Ο Πελάτης επιλέγει από το drop-down μενού πόσες θέσεις θέλει . | | |
| Γεγονός Εκκίνησης: | Ο Πελάτης εκτελεί την εφαρμογή Ticket | | |
| Προϋποθέσεις: | Είσοδος στο σύστημα | | |
| Τελική Κατάσταση: | Ο πελάτης επιλέγει από το drop-down μενού πόσα εισιτήρια χρειάζεται . | | |
| Φυσιολογική Ροή: | 1. UC - 8 (Είσοδος στο σύστημα) 2. Επιλογή πλήθους θέσεων. | | |
| Εναλλακτική Ροή: | 1. UC - 8 (Είσοδος στο σύστημα) 2. Επιλογή πλήθους θεσεων . | | |
|
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-8 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 30 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-3** | | |
| Ονομασία: | Επιλογή διαδρομής | | |
| Δημιουργήθηκε από: | ΜΑΡΙΝΑ ΑΡΝΤΙΤ | Τελευταία ενημέρωση από: | ΜΑΡΙΝΑ ΑΡΝΤΙΤ |
| Ημερομηνία Συγγραφής: | 12/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 22/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Πελάτης | | |
| Περιγραφή: | Ο Πελάτης επιλέγει την διαδρομή. | | |
| Γεγονός Εκκίνησης: | Ο Πελάτης εκτελεί την εφαρμογή. | | |
| Προϋποθέσεις: | Είσοδος στο σύστημα | | |
| Τελική Κατάσταση: | Ο πελάτης επιλέγει την διαδρομή | | |
| Φυσιολογική Ροή: | 1. UC-8 (Είσοδος στο σύστημα) 2. UC-2 (Εισαγωγή πλήθους θέσεων) 3. UC-4 (Εμφάνιση διαθέσιμων θέσεων) | | |
| Εναλλακτική Ροή: | 1. UC-8 (Είσοδος στο σύστημα) 2. UC-2 (Εισαγωγή πλήθους θέσεων) 3. UC-4 (Εμφάνιση διαθέσιμων θέσεων) | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-8, UC-2, UC-4 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 30 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-4** | | |
| Ονομασία: | Εμφάνιση διαθέσιμων θέσεων | | |
| Δημιουργήθηκε από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ | Τελευταία ενημέρωση από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ |
| Ημερομηνία Συγγραφής: | 10/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 14/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Σύστημα Κρατήσεων Εισιτηρίων | | |
| Περιγραφή: | Το σύστημα εμφανίζει τα διαθέσιμα εισιτήρια ανά το πλήθος ατόμων και την διαδρομή που επιλέχθηκε | | |
| Γεγονός Εκκίνησης: | Πατιέται το κουμπί Search | | |
| Προϋποθέσεις: | 1. Εκκίνηση της εφαρμογής 2. Πάτημα του κουμπιού Search | | |
| Τελική Κατάσταση: | Εμφάνιση διαθέσιμων θέσεων | | |
| Φυσιολογική Ροή: | 1. UC-8 (Είσοδος στο σύστημα) 2. UC-2 (Εισαγωγή πλήθους ατόμων) 3. UC-3 (Επιλογή διαδρομής ) 4. Πάτημα κουμπιού Search 5. Εμφάνιση διαθέσιμων θέσεων που τηρούν τα κριτήρια επιλογής | | |
| Εναλλακτική Ροή: | 1. UC-8 (Είσοδος στο σύστημα) 2. UC-3 (Επιλογή διαδρομής ) 3. Πάτημα κουμπιού Search 4. Εμφάνιση όλων των θέσεων τύπου επιλογής | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-2,UC-3, UC-8 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 50 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-5** | | |
| Ονομασία: | Εμφάνιση λεπτομερειών διαδρομών | | |
| Δημιουργήθηκε από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ | Τελευταία ενημέρωση από: | ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ |
| Ημερομηνία Συγγραφής: | 20/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 23/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Σύστημα κρατήσεων εισιτηρίων | | |
| Περιγραφή: | Το σύστημα εμφανίζει λεπτομέρειες για μη επιλεγμένη διαδρομή . | | |
| Γεγονός Εκκίνησης: | Πάτημα του κουμπιού Details… | | |
| Προϋποθέσεις: | 1. UC-8 (Είσοδος στο σύστημα) 2. Πάτημα του κουμπιού Search 3. Εμφάνιση διαθέσιμων θέσεων (UC-4) | | |
| Τελική Κατάσταση: | Εμφάνιση λεπτομερειών διαδρομών (διαθεσιμότητα) | | |
| Φυσιολογική Ροή: | 1. UC-4 (Εμφάνιση διαθέσιμων θέσεων ) 2. Επιλογή διαδρομής 3. Πάτημα κουμπιού Details… | | |
| Εναλλακτική Ροή: | - | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-3,UC-8 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 13 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

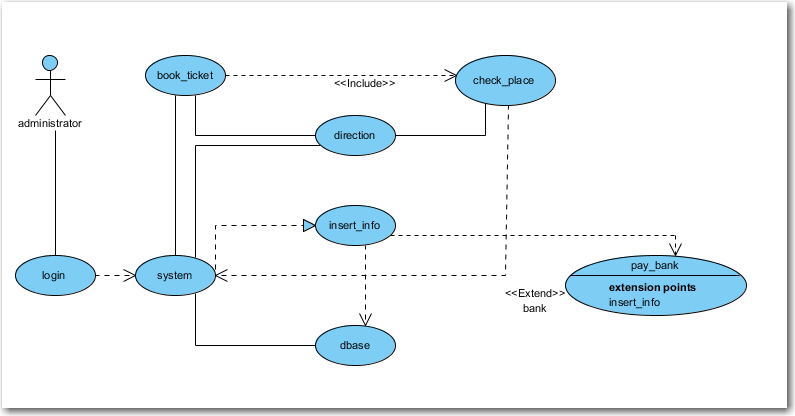
|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-6** | | |
| Ονομασία: | Εισαγωγή στοιχείων για κράτηση | | |
| Δημιουργήθηκε από: | ΜΑΡΙΝΑ ΑΡΝΤΙΤ | Τελευταία ενημέρωση από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ |
| Ημερομηνία Συγγραφής: | 25/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 23/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Πελάτης | | |
| Περιγραφή: | Ο Πελάτης εισάγει ονοματεπώνυμο και διαδρομή και ώρα αναχώρησης απο το σταθμο . | | |
| Γεγονός Εκκίνησης: | Πάτημα του κουμπιού Book | | |
| Προϋποθέσεις: | 1. UC-2 (Εισαγωγή πλήθους ατόμων) 2. UC-3 (Επιλογή τύπου διαδρομών ) 3. Πάτημα κουμπιού Search 4. UC-4 (Εμφάνιση διαθέσιμων θέσεων ) | | |
| Τελική Κατάσταση: | 1. Ο χρήστης εισάγει τα στοιχεία της κράτησης που επιθυμεί να κάνει | | |
| Φυσιολογική Ροή: | 1. UC-2 (Εισαγωγή πλήθους ατόμων) 2. UC-3 (Επιλογή τύπου διαδρομής ) 3. Πάτημα κουμπιού Search 4. UC-4 (Εμφάνιση διαθέσιμων θέσεων ) 5. Πάτημα κουμπιού Book 6. Εμφάνιση φόρμας εισαγωγής στοιχείων και οδηγίες εισαγωγής αυτών | | |
| Εναλλακτική Ροή: | - | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-2,UC-3, UC-4 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 8 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | -- | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-7** | | |
| Ονομασία: | Έλεγχος στοιχείων και κράτηση εισιτηρίων | | |
| Δημιουργήθηκε από: | ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ | Τελευταία ενημέρωση από: | ΚΩΝΣΤΑΝΤΑΡΑ ΑΡΤΕΜΙΣ |
| Ημερομηνία Συγγραφής: | 18/12/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 23/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Σύστημα Kρατήσεων Εισιτηρίων | | |
| Περιγραφή: | Το σύστημα ελέγχει αν τα στοιχεία που εισήγαγε ο πελάτης είναι ορθά και πραγματοποιεί την κράτηση ενημερώνοντας την βάση δεδομένων του. | | |
| Γεγονός Εκκίνησης: | 1. Ολοκλήρωση εισαγωγής στοιχείων(UC-6) 2. Πάτημα πλήκτρου ΟΚ | | |
| Προϋποθέσεις: | 1. UC-2 2. UC-3 3. UC-4 4. UC-6 5. Πάτημα πλήκτρου ΟΚ 6. Ύπαρξη βάσης δεδομένων και σύνδεση της με την εφαρμογή | | |
| Τελική Κατάσταση: | Η βάση δεδομένων ενημερώνει το συστήμα για την διαθεσιμότητα των θέσεων . | | |
| Φυσιολογική Ροή: | 1. UC-2 2. UC-3 3. UC-4 4. UC-6 5. Πάτημα πλήκτρου ΟΚ 6. Ενημέρωση βάσης δεδομένων | | |
| Εναλλακτική Ροή: | Δεν υπάρχει διαθεσιμότητα θέσεων (μη ενημέρωση της βάσης δεδομένων) | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | UC-2,UC-3, UC-4, UC-6 | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 35 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | - | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

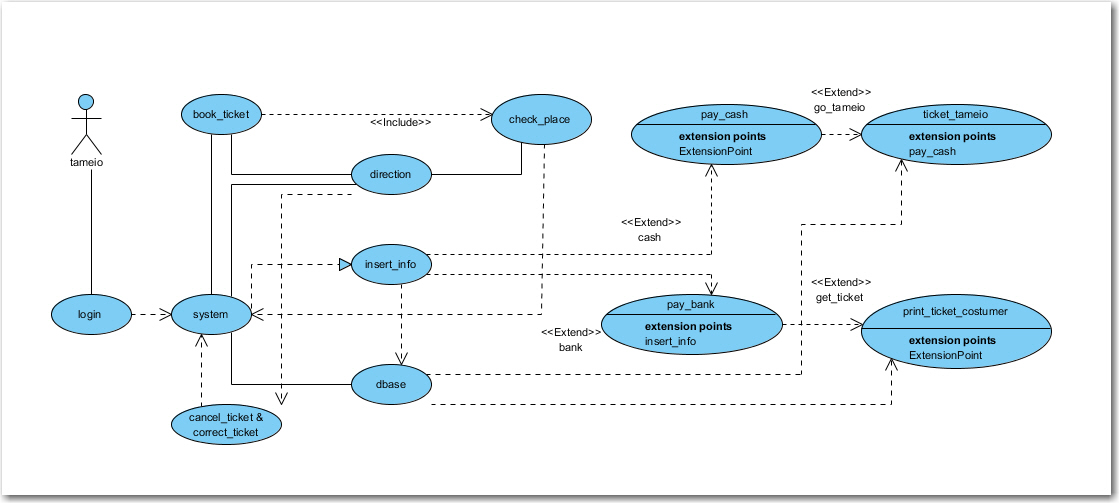
|  |  |  |  |
| --- | --- | --- | --- |
| Κωδικός Περίπτωσης: | **UC-8** | | |
| Ονομασία: | Είσοδος στο σύστημα | | |
| Δημιουργήθηκε από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ | Τελευταία ενημέρωση από: | ΜΕΤΣΗ ΣΙΛΒΙΤΑ |
| Ημερομηνία Συγγραφής: | 30/11/2012 | Ημερομηνία τελευταίας ενημέρωσης: | 25/01/2013 |
| Εμπλεκόμενοι Ρόλοι: | Πελάτης | | |
| Περιγραφή: | Είσοδος του πελάτη στο σύστημα. | | |
| Γεγονός Εκκίνησης: | Εκκίνηση της εφαρμογής Ticket | | |
| Προϋποθέσεις: | 1. Η εφαρμογή να είναι εκτελέσιμη | | |
| Τελική Κατάσταση: | Εμφάνιση επιλογών UC-2, UC-3, κουμπιού Search | | |
| Φυσιολογική Ροή: | Εκκίνηση της εφαρμογής Ticket | | |
| Εναλλακτική Ροή: | - | | |
| Εξαιρέσεις: | - | | |
| Ενσωματώνει: | - | | |
| Προτεραιότητα: | 1 | | |
| Συχνότητα χρήσης: | Περίπου 20 χρήσεις ανά εβδομάδα | | |
| Business Rules: | - | | |
| Ειδικές απαιτήσεις: | -- | | |
| Υποθέσεις: | - | | |
| Σημειώσεις και ζητήματα: | - | | |

**USER CASE DIAGRAM**

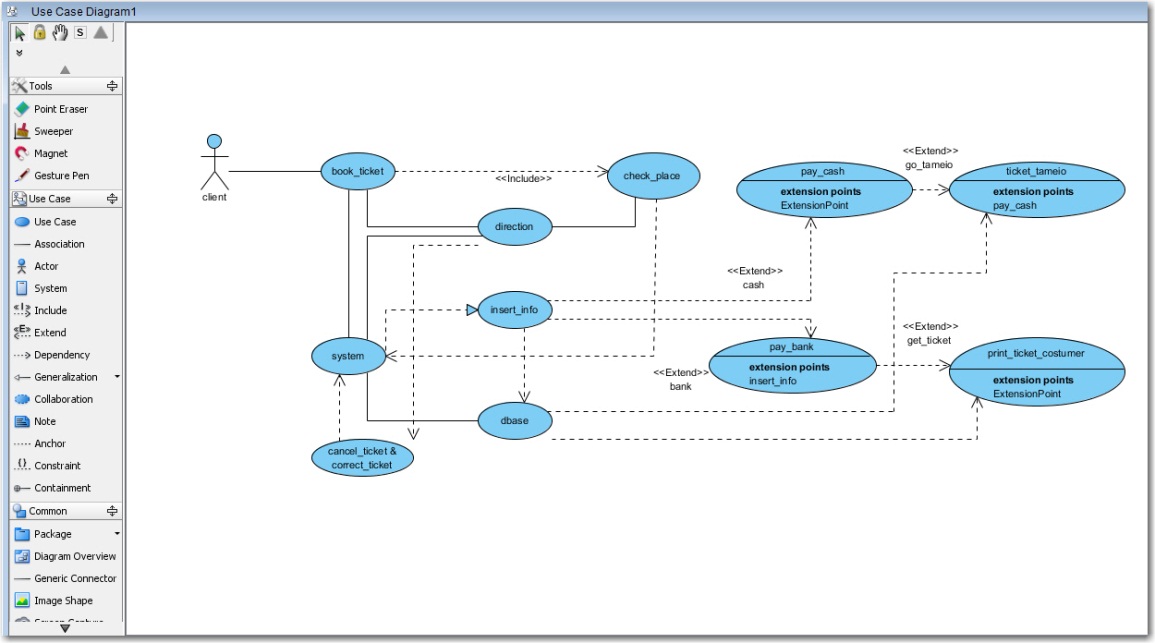
# ADMINISTRATOR



TAMIAS



# CLIENT



Kάρτες CRC + UML Διαγράμματα

για το

Σύστημα κρατήσεων εισιτηριων

Από TURBO

Version 1.0 approved

Δημιουργήθηκε από Ομάδα 59 (ΜΕΤΣΗ Σ., ΜΑΡΙΝΑ Α., ΚΩΝΣΤΑΝΤΑΡΑ Α.)

TURBO IT Solutions,

ΦΕΒΡΟΥΑΡΙΟΣ, 2013

# Κάρτες CRC-Διαγράμματα

* 1. **Απλοποιημένες Κάρτες CRC**

1. **Πελάτης.**
2. **Διαχειριστές** (Συμπεριλαμβάνονται : το προσωπικό της επιχειρησης.)
3. **Διαδρομές**

|  |  |
| --- | --- |
| **Κατηγορία: Πελάτης** | |
| **Αρμοδιότητες** | **Συνεργάτες** |
| **Ονοματεπώνυμο** |  |
| **Αριθμός Πιστωτικής Κάρτας (Τραπεζα)** |  |
|  |  |
| **Κράτηση θέσεων ()** | **Διαδρομές** |
| **Διόρθωση/Ακύρωση Κράτησης** | **Ταμίας , Πελάτης** |
| **Πληρωμή()** | **Ταμίας , Πελάτης** |

* Η Διόρθωση/Ακύρωση Κράτησης Εισιτηρίου απαιτεί συνεργασία με την οντότητα Σύστημα επειδή αυτός θα κάνει ουσιαστικά όταν ζητηθεί από τον Ταμία την Διόρθωση/Ακύρωση της Κράτησης. Η Κράτηση Εισιτηρίου από την πλευρά του Πελάτη συνεργάζεται με το σύστημα οπότε μπορούμε να πούμε ότι δεν εμπλέκεται η οντότητα Ταμίας, αν και αν πούμε ότι συνεργάζεται δεν νομίζω ότι είναι και τόσο λάθος…Αν όμως θέσουμε οντότητα Εισιτήριο τότε θα πρέπει να επικοινωνεί ο πελάτης με αυτήν για να δει την διαθεσιμότητά τους

|  |  |
| --- | --- |
| **Κατηγορία: Ταμίας** | |
| **Αρμοδιότητες** | **Συνεργάτες** |
| **Ονοματεπώνυμο** |  |
|  |  |
|  |  |
| **Κράτηση θέσεων ()** | **Διαδρομές** |
| **Διόρθωση/Ακύρωση Κράτησης** | **Ταμίας , Πελάτης** |
| **Πληρωμή()** | **Ταμίας , Πελάτης** |

|  |  |
| --- | --- |
| **Κατηγορία: Διαχειριστές** | |
| **Αρμοδιότητες** | **Συνεργάτες** |
| **Λίστα Διαδρομών** |  |
| **Λίστα θέσεων** |  |
| **ΙD Υπαλλήλου** |  |
| **Υπολογισμός κόστους()** | **Διαδρομές** |

* Οι διαχειριστές ανά πάσα στιγμή πρέπει να γνωρίζουν ποιές είναι οι διαδρομές, (λίστα διαδρομών), ποιές είναι οι θέσεις (λίστα θέσεων), πόσο είναι το Συνολικό Κόστος Διαδρομής που πρέπει να πληρώσει ο Πελάτης**.** Στην λειτουργία αυτή συμπεριλαμβάνεται η αλλαγή της τιμής ενός εισιτηρίου ή η αλλαγή της ώρας ή ακομα και η διαγραφή μια κράτησης .

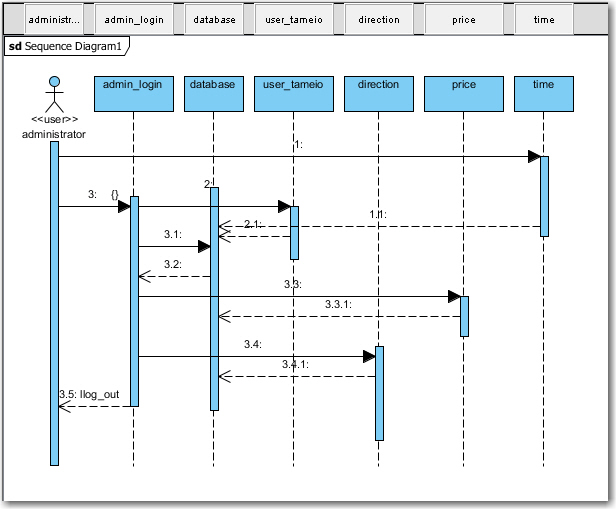
|  |  |
| --- | --- |
| **Κατηγορία: Διαδρομή** | |
| **Αρμοδιότητες** | **Συνεργάτες** |
| **Κωδικός Διαδρομής** |  |
| **Τιμή Διαδρομής** |  |
| **Χωρητικότητα (αριθμός θέσεων )** |  |
| **Διαθεσιμότητα** |  |
| **Ώρα Κράτησης** |  |
| **Στοιχεία Πελάτη()** | **Πελάτης** |

* Η Διαδρομή θα πρέπει να γνωρίζει την τίμη του,τον μοναδικό κωδικό που της έχει ανατεθεί,την μέγιστη χωρητικότητα θέσεων , την διαθεσιμότητα της, καθώς και να γνωρίζει ποτέ είναι η αναχώρηση.

# UML Διαγράμματα

# SEQUENCE DIAGRAM

# ADMINISTRATOR



**CLIENT**

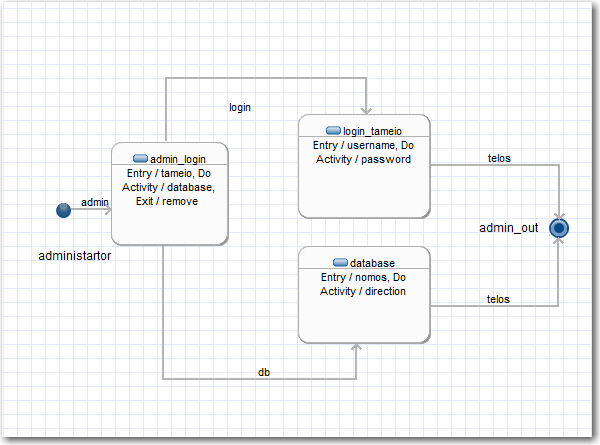
# 

# TAMIAS

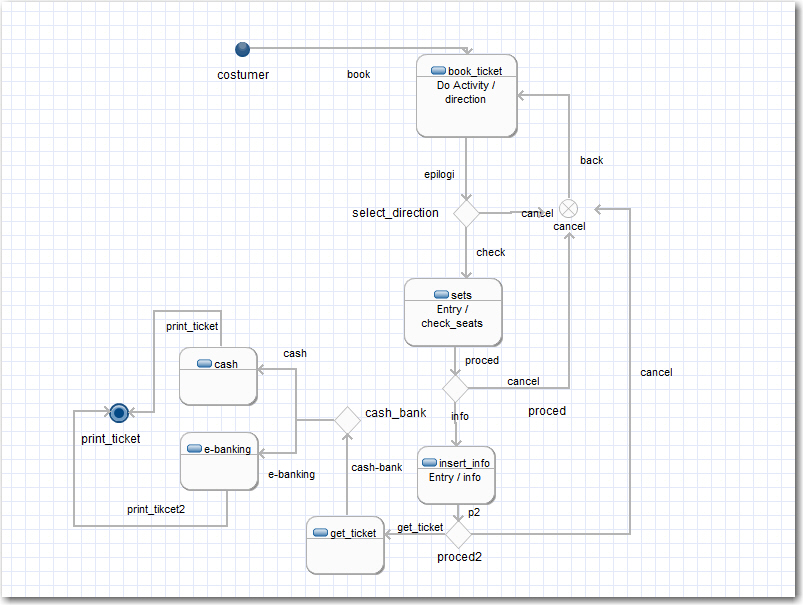
# 

# STATE DIAGRAM

**ADMINISTRATOR**



# CLIENT



TAMIAS

# 

# 8.0 ΚΩΔΙΚΑΣ ΚΛΑΣΕΩΝ

**8.1 AΡΧΙΚΟΣ KΩΔΙΚΑΣ ΚΛΑΣΗΣ Client:**

import java.sql.\*;

import java.lang.Exception;

import javax.swing.ImageIcon;

import com.itextpdf.text.Document;

import com.itextpdf.text.Paragraph;

import com.itextpdf.text.pdf.PdfPTable;

import com.itextpdf.text.pdf.PdfWriter;

import java.awt.Toolkit;

import java.io.\*;

import javax.swing.JOptionPane;

public class Client extends javax.swing.JFrame {

public Client() {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jLabel6 = new javax.swing.JLabel();

jCheckBox1 = new javax.swing.JCheckBox();

jCheckBox2 = new javax.swing.JCheckBox();

Name = new javax.swing.JTextField();

Surname = new javax.swing.JTextField();

Spot = new javax.swing.JTextField();

Location = new javax.swing.JTextField();

Age = new javax.swing.JTextField();

jButton1 = new javax.swing.JButton();

Time = new javax.swing.JTextField();

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

setTitle("ΞΞ‘Ξ¤Ξ‘Ξ§Ξ©Ξ΅Ξ—Ξ£Ξ— Ξ Ξ•Ξ›Ξ‘Ξ¤Ξ—");

setResizable(false);

jLabel1.setText("ONOMA");

jLabel2.setText("Ξ•Ξ Ξ©ΞΞ¥ΞΞ");

jLabel3.setText("Ξ—Ξ›Ξ™ΞΞ™Ξ‘");

jLabel4.setText("ΞΞ•Ξ£Ξ—");

jLabel5.setText("Ξ”Ξ™Ξ‘Ξ”Ξ΅ΞΞΞ—");

jLabel6.setText("Ξ©Ξ΅Ξ‘");

jCheckBox1.setText("ΞΞ•Ξ¤Ξ΅Ξ—Ξ¤Ξ‘");

jCheckBox2.setText("Ξ¤Ξ΅Ξ‘Ξ Ξ•Ξ–Ξ‘Ξ£");

jButton1.setText("ΞΞ");

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

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

jButton1ActionPerformed(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)

.addComponent(jLabel1)

.addComponent(jLabel2)

.addComponent(jLabel3)

.addComponent(jLabel4)

.addComponent(jLabel5)

.addComponent(jLabel6))

.addGap(41, 41, 41)

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

.addComponent(jButton1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addComponent(jCheckBox1)

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

.addComponent(jCheckBox2))

.addComponent(Name)

.addComponent(Surname)

.addComponent(Spot)

.addComponent(Location)

.addComponent(Age)

.addComponent(Time))

.addContainerGap(57, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

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

.addComponent(jLabel1)

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

.addGap(18, 18, 18)

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

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

.addComponent(jLabel2))

.addGap(23, 23, 23)

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

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

.addComponent(jLabel3))

.addGap(26, 26, 26)

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

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

.addComponent(jLabel4))

.addGap(29, 29, 29)

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

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

.addComponent(jLabel5))

.addGap(27, 27, 27)

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

.addComponent(jLabel6)

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

.addGap(29, 29, 29)

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

.addComponent(jCheckBox1)

.addComponent(jCheckBox2))

.addGap(18, 18, 18)

.addComponent(jButton1)

.addContainerGap(61, Short.MAX\_VALUE))

);

java.awt.Dimension screenSize = java.awt.Toolkit.getDefaultToolkit().getScreenSize();

setBounds((screenSize.width-328)/2, (screenSize.height-446)/2, 328, 446);

}// </editor-fold>//GEN-END:initComponents

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

String MessageCash="Go Pay";

String MessageBank="Enter Card Details";

Document document = new Document();

if (jCheckBox1.isSelected())

{

try {

String str1=Name.getText();

String str2=Surname.getText();

String str3=Age.getText();

String str4=Spot.getText();

String str5=Location.getText();

String str6=Time.getText();

PdfWriter.getInstance(document, new FileOutputStream("Ticket.pdf"));

document.open();

document.add(new Paragraph("TicketHouse"));

PdfPTable table=new PdfPTable(2);

table.addCell("Name");

table.addCell(str1);

table.addCell("Surname");

table.addCell(str2);

table.addCell("Age");

table.addCell(str3);

table.addCell("Spot");

table.addCell(str4);

table.addCell("Location");

table.addCell(str5);

table.addCell("Time");

table.addCell(str6);

document.add(table);

document.close();

}

catch(Exception e){

}

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(null,MessageCash);

}

else if (jCheckBox2.isSelected())

{

try {

String str1=Name.getText();

String str2=Surname.getText();

String str3=Age.getText();

String str4=Spot.getText();

String str5=Location.getText();

String str6=Time.getText();

Toolkit.getDefaultToolkit().beep();

String input = JOptionPane.showInputDialog("Enter CardNumber:");

PdfWriter.getInstance(document, new FileOutputStream("Ticket.pdf"));

document.open();

document.add(new Paragraph("TicketHouse"));

PdfPTable table=new PdfPTable(2);

table.addCell("Name");

table.addCell(str1);

table.addCell("Surname");

table.addCell(str2);

table.addCell("Age");

table.addCell(str3);

table.addCell("Spot");

table.addCell(str4);

table.addCell("Location");

table.addCell(str5);

table.addCell("Time");

table.addCell(str6);

table.addCell("CardNumber");

table.addCell(input);

document.add(table);

document.close();

}

catch(Exception e){

}

}

}//GEN-LAST:event\_jButton1ActionPerformed

public static void main(String args[]) {

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

public void run() {

new Client().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JTextField Age;

private javax.swing.JTextField Location;

private javax.swing.JTextField Name;

private javax.swing.JTextField Spot;

private javax.swing.JTextField Surname;

private javax.swing.JTextField Time;

private javax.swing.JButton jButton1;

private javax.swing.JCheckBox jCheckBox1;

private javax.swing.JCheckBox jCheckBox2;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLabel jLabel6;

// End of variables declaration//GEN-END:variables }

**8.2 AΡΧΙΚΟΣ KΩΔΙΚΑΣ ΚΛΑΣΗΣ Home :**

import java.sql.\*;

import java.text.MessageFormat;

import javax.swing.JTable;

import java.lang.Exception;

import javax.swing.ImageIcon;

import java.sql.ResultSet;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JOptionPane;

import net.proteanit.sql.DbUtils;

public class Home extends javax.swing.JFrame {

public Home() {

initComponents();

}

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jButton1 = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTable1 = new javax.swing.JTable();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jButton4 = new javax.swing.JButton();

jMenuBar1 = new javax.swing.JMenuBar();

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

setTitle("Ξ•ΞΞ”ΞΞ¤Ξ—Ξ΅Ξ™Ξ");

jButton1.setText("ΞΞ‘Ξ¤Ξ‘Ξ§Ξ©Ξ΅Ξ—Ξ£Ξ— Ξ£Ξ¤ΞΞ™Ξ§Ξ•Ξ™Ξ©Ξ");

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

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

jButton1ActionPerformed(evt);

}

});

jTable1.setModel(new javax.swing.table.DefaultTableModel(

new Object [][] {

},

new String [] {

"Ξ©Ξ΅Ξ‘Ξ΅Ξ™Ξ", "Ξ Ξ΅ΞΞ£Ξ¦ΞΞ΅Ξ•Ξ£"

}

) {

boolean[] canEdit = new boolean [] {

false, false

};

public boolean isCellEditable(int rowIndex, int columnIndex) {

return canEdit [columnIndex];

}

});

jScrollPane1.setViewportView(jTable1);

jButton2.setText("Ξ•Ξ¥Ξ’ΞΞ™Ξ‘");

jButton2.addActionListener(new java.awt.event.ActionListener() {

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

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Ξ›Ξ‘ΞΞ™Ξ‘");

jButton3.addActionListener(new java.awt.event.ActionListener() {

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

jButton3ActionPerformed(evt);

}

});

jButton4.setText("Ξ Ξ‘Ξ¤Ξ΅Ξ‘");

jButton4.addActionListener(new java.awt.event.ActionListener() {

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

jButton4ActionPerformed(evt);

}

});

setJMenuBar(jMenuBar1);

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

.addGap(20, 20, 20)

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE)

.addGroup(layout.createSequentialGroup()

.addComponent(jButton2)

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

.addComponent(jButton3)

.addGap(18, 18, 18)

.addComponent(jButton4))

.addComponent(jButton1, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

.addContainerGap(39, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

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

.addGroup(layout.createSequentialGroup()

.addContainerGap()

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

.addComponent(jButton2)

.addComponent(jButton3)

.addComponent(jButton4))

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

.addComponent(jButton1)

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

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, 181, javax.swing.GroupLayout.PREFERRED\_SIZE)

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

);

java.awt.Dimension screenSize = java.awt.Toolkit.getDefaultToolkit().getScreenSize();

setBounds((screenSize.width-302)/2, (screenSize.height-328)/2, 302, 328);

}// </editor-fold>//GEN-END:initComponents

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

new Client().setVisible(true);

}//GEN-LAST:event\_jButton1ActionPerformed

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton2ActionPerformed

try{

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

String sourceURL=new String("jdbc:mysql://localhost:3306/test");

java.sql.Connection ConnectiondatabaseConnection = DriverManager.getConnection(sourceURL);

Statement statement=ConnectiondatabaseConnection.createStatement();

String sql="SELECT \* FROM ktel\_euboias";

ResultSet result=statement.executeQuery(sql);

jTable1.setModel(DbUtils.resultSetToTableModel(result));

ConnectiondatabaseConnection.close();

}

catch(Exception e1){

System.out.println(e1);

}

}//GEN-LAST:event\_jButton2ActionPerformed

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton3ActionPerformed

try{

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

String sourceURL=new String("jdbc:mysql://localhost:3306/test");

java.sql.Connection ConnectiondatabaseConnection = DriverManager.getConnection(sourceURL);

Statement statement=ConnectiondatabaseConnection.createStatement();

String sql="SELECT \* FROM ktel\_lamias";

ResultSet result=statement.executeQuery(sql);

jTable1.setModel(DbUtils.resultSetToTableModel(result));

ConnectiondatabaseConnection.close();

}

catch(Exception e1){

System.out.println(e1);

}

}//GEN-LAST:event\_jButton3ActionPerformed

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton4ActionPerformed

try{

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

String sourceURL=new String("jdbc:mysql://localhost:3306/test");

java.sql.Connection ConnectiondatabaseConnection = DriverManager.getConnection(sourceURL);

Statement statement=ConnectiondatabaseConnection.createStatement();

String sql="SELECT \* FROM ktel\_patras";

ResultSet result=statement.executeQuery(sql);

jTable1.setModel(DbUtils.resultSetToTableModel(result));

ConnectiondatabaseConnection.close();

}

catch(Exception e1){

System.out.println(e1);

}

}//GEN-LAST:event\_jButton4ActionPerformed

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(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

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

} catch (IllegalAccessException ex) {

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

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Home.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 Home().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;

private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JMenuBar jMenuBar1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JTable jTable1;

// End of variables declaration//GEN-END:variables

}

**8.3 AΡΧΙΚΟΣ KΩΔΙΚΑΣ ΚΛΑΣΗΣ GUI :**

import java.awt.Toolkit;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.awt.event.WindowEvent;

import java.io.DataInputStream;

import java.util.Random;

import javax.swing.JOptionPane;

public class Gui extends javax.swing.JFrame

{

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

private void initComponents() {

jPanel1 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

jPasswordField1 = new javax.swing.JPasswordField();

jButton1 = new javax.swing.JButton();

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

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jLabel1.setText("Username");

jLabel2.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N

jLabel2.setText("Password");

jTextField1.addKeyListener(new java.awt.event.KeyAdapter() {

public void keyPressed(java.awt.event.KeyEvent evt) {

jTextField1KeyPressed(evt);

}

});

jPasswordField1.addKeyListener(new java.awt.event.KeyAdapter() {

public void keyPressed(java.awt.event.KeyEvent evt) {

jPasswordField1KeyPressed(evt);

}

});

jButton1.setText("OK");

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

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

jButton1ActionPerformed(evt);

}

});

jButton1.addKeyListener(new java.awt.event.KeyAdapter() {

public void keyPressed(java.awt.event.KeyEvent evt) {

jButton1KeyPressed(evt);

}

});

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

jPanel1.setLayout(jPanel1Layout);

jPanel1Layout.setHorizontalGroup(

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

.addGroup(jPanel1Layout.createSequentialGroup()

.addContainerGap()

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

.addGroup(jPanel1Layout.createSequentialGroup()

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

.addGap(178, 178, 178))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

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

.addGap(179, 179, 179))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.createSequentialGroup()

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

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

.addComponent(jPasswordField1, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jTextField1, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton1, javax.swing.GroupLayout.DEFAULT\_SIZE, 101, Short.MAX\_VALUE))

.addGap(57, 57, 57))))

);

jPanel1Layout.setVerticalGroup(

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

.addGroup(jPanel1Layout.createSequentialGroup()

.addGap(19, 19, 19)

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

.addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 20, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

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

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

.addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED\_SIZE, 20, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGap(29, 29, 29)

.addComponent(jButton1)

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

);

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

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

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

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED\_SIZE, 220, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGap(0, 3, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

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

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

);

java.awt.Dimension screenSize = java.awt.Toolkit.getDefaultToolkit().getScreenSize();

setBounds((screenSize.width-239)/2, (screenSize.height-175)/2, 239, 175);

}// </editor-fold>//GEN-END:initComponents

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event\_jButton1ActionPerformed

String username = jTextField1.getText();

String password = jPasswordField1.getText();

String Message = "Please Enter Valid Information";

if (username.equals("ardit") && password.equals("1234"))

{

new Home().setVisible(true);

setVisible(false);

dispose();

}

else

{

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(null,Message);

}

}//GEN-LAST:event\_jButton1ActionPerformed

private void jButton1KeyPressed(java.awt.event.KeyEvent evt) {//GEN-FIRST:event\_jButton1KeyPressed

String username = jTextField1.getText();

String password = jPasswordField1.getText();

String Message = "Please Enter Valid Information";

int key = evt.getKeyCode();

if (key == KeyEvent.VK\_ENTER)

{

if (username.equals("ardit") && password.equals("1234"))

{

new Home().setVisible(true);

setVisible(false);

dispose();

}

else

{

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(null,Message);

}

}

}//GEN-LAST:event\_jButton1KeyPressed

private void jPasswordField1KeyPressed(java.awt.event.KeyEvent evt) {//GEN-FIRST:event\_jPasswordField1KeyPressed

String username = jTextField1.getText();

String password = jPasswordField1.getText();

String Message = "Please Enter Valid Information";

int key = evt.getKeyCode();

if (key == KeyEvent.VK\_ENTER)

{

if (username.equals("ardit") && password.equals("1234"))

{

new Home().setVisible(true);

setVisible(false);

dispose();

}

else

{

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(null,Message);

}

}

}//GEN-LAST:event\_jPasswordField1KeyPressed

private void jTextField1KeyPressed(java.awt.event.KeyEvent evt) {//GEN-FIRST:event\_jTextField1KeyPressed

String username = jTextField1.getText();

String password = jPasswordField1.getText();

String Message = "Please Enter Valid Information";

int key = evt.getKeyCode();

if (key == KeyEvent.VK\_ENTER)

{

if (username.equals("root") && password.equals("anik"))

{

new Home().setVisible(true);

setVisible(false);

dispose();

}

else

{

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(null,Message);

}

}

}//GEN-LAST:event\_jTextField1KeyPressed

public static void main(String args[])

{

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

public void run() {

new Gui().setVisible(true);

}

});

}

// Variables declaration - do not modify//GEN-BEGIN:variables

private javax.swing.JButton jButton1;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JPanel jPanel1;

private javax.swing.JPasswordField jPasswordField1;

private javax.swing.JTextField jTextField1;

// End of variables declaration//GEN-END:variables

public Gui()

{

super();

create();

this.setVisible(true);

initComponents();

}

private void create()

{

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

jPasswordField1 = new javax.swing.JPasswordField();

jButton1 = new javax.swing.JButton();

jPanel1 = (javax.swing.JPanel)this.getContentPane();

}

# 9.0 INTERFACE

# Ο ταμίας θα πραγματοποιεί το παρακάτω log in για να ξεκινήσει η βάρδια του.

# 

# Ο πελάτης θα βλέπει το συγκεκριμένο παράθυρο για να επιλέξει την διαδρομή του.

# 

# Εδώ θα παρουσιάζονται τα ωράρια των δρομολογίων και οι προσφορές που μπορεί να υπάρχουν.

# 

* Στην συνέχεια εμφανίζεται το παράθυρο για να εισάγει τα στοιχεία του ο πελάτης.

# 

# Αφού εισαχθούν τα στοιχεία όπως παρακάτω, επιλέγει ο πελάτης τον τρόπο πληρωμής. Εάν επιθυμεί να πληρώσει με μετρητά «κλικάρει» το box μετρητά

# 

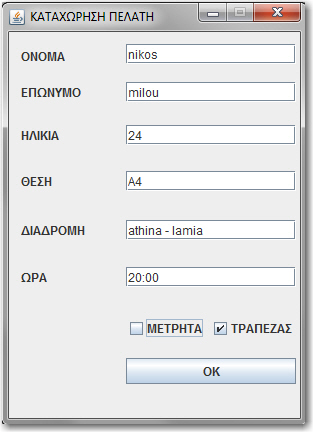
* Μήνυμα που δηλώνει ότι τα στοιχεία του πελάτη καταχωρήθηκαν και πρέπει να προχωρήσει στον ταμία να πληρώσει με μετρητά

# 

* Στην συνέχεια ο ταμίας εκτυπώνει το εισιτήριο.

# 

# Εάν όμως επιθυμεί να πληρώσει με πιστωτική κάρτα τότε θα πρέπει να «κλικάρει» την επιλογή για ΤΡΑΠΕΖΑ.



# Θα εμφανιστεί παράθυρο για εισαγωγή του αριθμού πιστωτικής κάρτας.

# 

# Ο πελάτης θα εισάγει τον αριθμό της πιστωτικής του κάρτας.

# 

# Και τέλος θα εκτυπωθεί το εισιτήριο.

# 

# 9.0 CODE COVERAGE

**Client-test(JUNIT)**

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

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.\*;

/\*\*

\* @author samsun

\*/

public class ClientTest {

public ClientTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class Client.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

Client.main(args);

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

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

}

}

# Home-Test(JUNIT)

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

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.\*;

/\*\*

\*

\* @author samsun

\*/

public class HomeTest {

public HomeTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class Home.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

Home.main(args);

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

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

}

}

# GUI-Test(JUNIT)

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

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.\*;

/\*\*

\*

\* @author samsun

\*/

public class GuiTest {

public GuiTest() {

}

@BeforeClass

public static void setUpClass() {

}

@AfterClass

public static void tearDownClass() {

}

@Before

public void setUp() {

}

@After

public void tearDown() {

}

/\*\*

\* Test of main method, of class Gui.

\*/

@Test

public void testMain() {

System.out.println("main");

String[] args = null;

Gui.main(args);

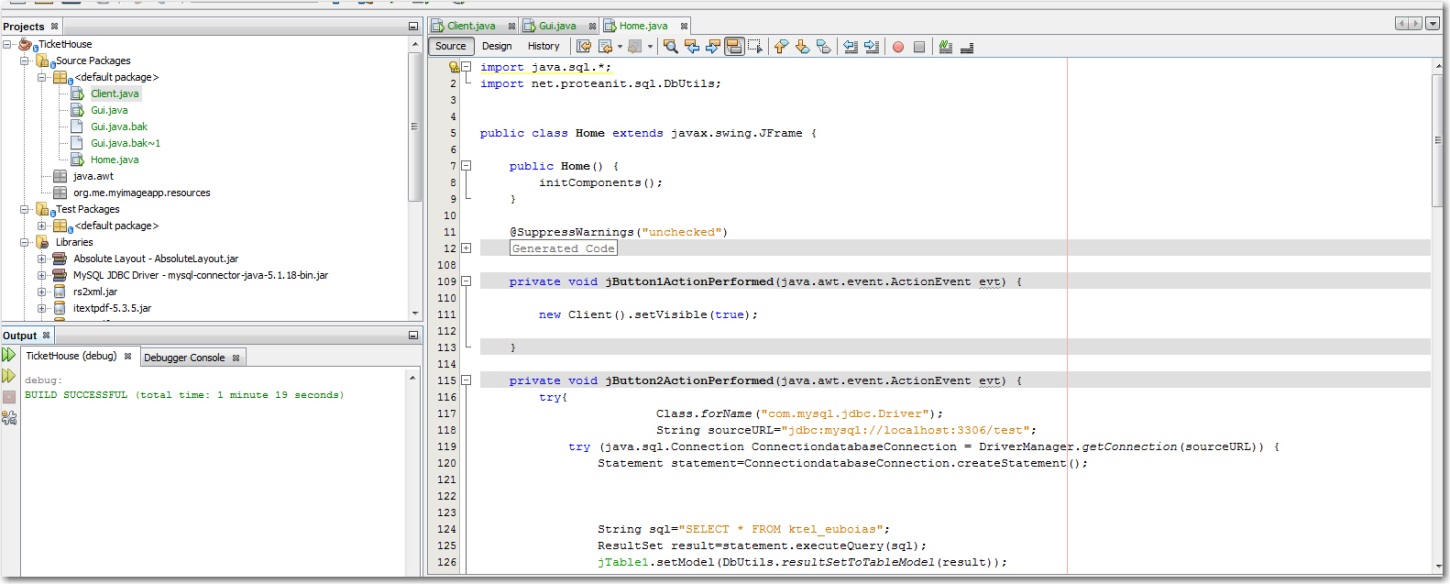
// TODO review the generated test code and remove the default call to fail.

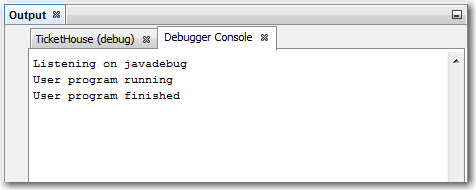
fail("The test case is a prototype.");

}

}

# 9.1 DEBBUGING





# . ΓΕΝΙΚΑ

* **ΤΕΛΙΚΟ ΠΡΟΓΡΑΜΜΑ ΣΕ NETBEANS**

*Το πρόγραμμα του της έκδοσης εισιτηρίων αποτελείται από 3 κλάσεις τις Client, Home και Gui, κάθε μια από τις οποίες έχει το δικό της interface. Η βασική ιδέα είναι ο πελάτης να επιλέγει τι την διαδρομή που επιθυμεί και για πόσα άτομα και το πρόγραμμα να του εμφανίζει αν υπάρχουν διαθέσιμες θέσεις και προσφορές ανάλογα με τις επιλογές του (οι διαδρομές και οι πληροφορίες για την κάθε μια είναι αποθηκευμένα σε μια βάση δεδομένων). Στη συνέχεια ο χρήστης θα μπορεί να κάνει κράτηση σε όποια διαδρομή επιθυμεί αλλά και να επιλέξει αριθμό θέσεων.*