Χαροκόπειο Πανεπιστήμιο
Τμήμα Πληροφορικής & Τηλεματικής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Συστήματα διαχείρισης στόλου, σχεδίαση και ανάπτυξη λογισμικού για την διαχείριση πληρώματος πλοίων

Πράγιας Ιωάννης

Α.Μ. – 20826

Αθήνα, Φεβρουάριος 2014

Χαροκόπειο Πανεπιστήμιο
Τμήμα Πληροφορικής & Τηλεματικής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Συστήματα διαχείρισης στόλου, σχεδίαση και ανάπτυξη λογισμικού για την διαχείριση πληρώματος πλοίων

Πράγιας Ιωάννης

Α.Μ. – 20826

Επιβλέπων καθηγητής

Δημητρακόπουλος Γεώργιος, Λέκτορας

Μέλη της εξεταστικής επιτροπής

Αναγνωστόπουλος Δημοσθένης, Καθηγητής

Δημητρακόπουλος Γεώργιος, Λέκτορας

Τσερπές Κωνσταντίνος, Λέκτορας

Πρόλογος

Η εκπόνηση της παρούσας πτυχιακής εργασίας πραγματοποιήθηκε στα πλαίσια φοίτησης στο Τμήμα Πληροφορικής και Τηλεματικής του Χαροκοπείου Πανεπιστημίου Αθηνών από τον Νοέμβριο του 2013 έως τον Φεβρουάριο του 2014.

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της πτυχιακής εργασίας, κο. Δημητρακόπουλο Γεώργιο για την υποστήριξη και την συνεχή καθοδήγηση που μου προσέφερε ως το πέρας της εργασίας.

Εν συνεχεία, θα ήθελα να ευχαριστήσω την οικογένεια μου που με στήριξε και συνεχίζει να με στηρίζει σε κάθε μου βήμα, τους συμφοιτητές και φίλους μου Μακρή Αντώνη, Βολάκη Γιώργο, Φωκέα Αέτιο, Μοσχάτο Γιώργο, Σπίθα Ευαγγελία και την Παπακανέλλου Νικολέτα .

Τέλος, δεν θα μπορούσα να παραλείψω, να ευχαριστήσω δύο ακόμα φίλους και ανθυποπλοίαρχους του εμπορικού ναυτικού Βενιζέλο Γιώργο και Στυλιανού Ηλία για τις πολύτιμες πληροφορίες που μου παρείχαν για τη σχεδίαση της εφαρμογής.

Περίληψη

Η άνθηση των επιστημών της πληροφορικής και των τηλεπικοινωνιών που βιώνουμε καθημερινά στις μέρες μας, έχει φέρει σημαντικές αλλαγές σε κάθε τομέα της δραστηριότητας των ατόμων. Δεν θα μπορούσε άλλως τε να είναι τυχαίο, ότι η εποχή που διανύουμε χαρακτηρίζεται ως εποχή της πληροφορίας. Σε μια τέτοια εποχή θα ήταν αδύνατο μια τόσο σημαντική βιομηχανία όπως είναι η ναυτιλιακή βιομηχανία να μείνει ανεπηρέαστη από τις εξελίξεις.

Οι ραγδαίες εξελίξεις των τελευταίων ετών στο χώρο της τηλεματικής, συνέβαλαν στην ανάπτυξη συστημάτων διαχείρισης στόλου οχημάτων τα οποία επέτρεψαν στις εταιρίες διανομών να επιτηρούν το στόλο τους σε πραγματικό χρόνο και να βελτιώσουν την εκτέλεση των δρομολογίων αντιμετωπίζοντας αρκετές απρόβλεπτες καταστάσεις. Πιο συγκεκριμένα η παρούσα πτυχιακή εργασία αναφέρεται σε συστήματα διαχείρισης στόλου. Καθώς επίσης και στην ανάπτυξη πληροφοριακού συστήματος με σκοπό την οργάνωση των χαρακτηριστικών στοιχείων του πλοίου, των προσωπικών δεδομένων του πληρώματος και την αυτοματοποίηση της διαδικασίας παραγωγής χρήσιμων εγγράφων. Αρχικά παρουσιάζονται τα χαρακτηριστικά στοιχεία, οι λειτουργίες και η αρχιτεκτονική συστημάτων διαχείρισης στόλου καθώς και σύγχρονα συστήματα διαχείρισης σε πραγματικό χρόνο. Ακόμα αναφέρεται η διαδικασία διαχείρισης στόλου πλοίων και η διαδικασία δρομολόγησης για την μεταφορά φορτίων. Τέλος πραγματοποιείται ανάλυση για τα υπάρχοντα ναυτιλιακά συστήματα της αγοράς.

Το σύστημα που κατασκευάστηκε στα πλαίσια της εν λόγω πτυχιακής εργασίας είναι ένα πληροφοριακό σύστημα, με εφαρμογή σε ηλεκτρονικούς υπολογιστές. Η εφαρμογή επιτρέπει την οργάνωση και την επεξεργασία των δεδομένων του πλοίου, του πληρώματος και την δημιουργία εγγράφων. Ειδικότερα με την χρήση του συστήματος αυτού, παρέχεται δυνατότητα στον χρήστη να προσθέσει, να επεξεργαστεί και να διαγράψει προσωπικά δεδομένα των μελών του πληρώματος καθώς και την επεξεργασία δεδομένων του πλοίου. Επίσης, το σύστημα που

υλοποιήθηκε προσφέρει την δυνατότητα παραγωγής εγγράφων τύπου xls αυτοματοποιημένα. Τα έγγραφα αυτά έχουν άμεση σχέση με τα δεδομένα του πλοίου, του πληρώματος και είναι απαραίτητα για την άφιξη του πλοίου στα λιμάνια προορισμού του.

Abstract

The development of computer and telecommunications science that we daily experience nowadays, has brought important changes in every sector of individuals' activity. It couldn't be a coincidence that the modern world is characterized as the information age. At such a time it would be impossible for a major industry such as the shipping industry to remain unaffected by these developments.

The rapid developments of the last years in the field of telematics, contributed to the development of the fleet management systems that allowed the distribution companies to monitor their fleet in real time and improve the performance of services by addressing several unforeseen situations. More specifically, the present study indicates the fleet management systems, as well as the development of information systems for the organization of the characteristics of the ship, the crew's personal data and to automate the process of producing useful documents. Initially the features, functions and architecture of the fleet management systems are presented as well as management modern systems in real time. In addition, the process management fleet of ships and routing procedure for cargo carriage is mentioned. Finally an analysis on existing maritime systems market is performed.

The system that was constructed as part of this study is an information system, with implement on computers. The application allows the organization and data processing of the ship and its crew and creates documents. More specifically the use of the system, provides the user with the ability to add, edit and delete personal data of crew members as well as the ship's data processing. Moreover the system that was developed offers the ability of producing xls type documents automatically. These documents are directly related to the data of the ship ,it's crew and are essential for the harbor destinations of the ship.

Περιεχόμενα

Πίνακας περιεχομένων

Κεφάλαιο 1

Εισαγωγή

Η ραγδαία ανάπτυξη της πληροφορικής, των τηλεπικοινωνιών και του διαδικτύου που βιώνουμε στις μέρες μας, έχει οδηγήσει στην διείσδυση των Τεχνολογιών της Πληροφορίας και Επικοινωνιών (ΤΠΕ) σε κάθε τομέα της δραστηριότητας των ανθρώπων.

Η ναυτιλιακή βιομηχανία αποτελεί μία από τις αρχαιότερες μορφές βιομηχανίας της ανθρωπότητας. Αξίζει να σημειωθεί ότι μέχρι και σήμερα η ναυτιλία παραμένει μία από τις σημαντικότερες βιομηχανίες σε παγκόσμιο επίπεδο. Σημαντικός παράγοντας για την επίτευξη των στόχων της αποτελεί η ικανότητά της να προσαρμόζεται στα νέα δεδομένα, και να χρησιμοποιεί σύγχρονα τεχνολογικά επιτεύγματα με σκοπό την μεγιστοποίηση της αποδοτικότητας, της ασφάλειας αλλά και της ανάπτυξης της.

Σκοπός της παρούσας πτυχιακής εργασίας είναι η θεωρητική ανάλυση της διαχείρισης στόλου και η ανάπτυξη λογισμικού για την διαχείριση πληρώματος πλοίων. Συγκεκριμένα το λογισμικό που αναπτύχθηκε αποσκοπεί στην οργάνωση των δεδομένων του πλοίου και του πληρώματος του, καθώς και στην αυτοματοποιημένη παραγωγή βασικών και χρήσιμων εγγράφων για την άφιξη των πλοίων στα λιμάνια προορισμού τους. Στο σημείο αυτό πρέπει να σημειωθεί ότι ως και σήμερα για την οργάνωση των δεδομένων του πλοίου, του πληρώματος και για την παραγωγή των εγγράφων δεν χρησιμοποιείται κάποιο λογισμικό. Δηλαδή, όλη η οργάνωση των δεδομένων πραγματοποιείται από τους αξιωματικούς του πλοίου μέσω εγγράφων τύπου xls. Πρέπει να αναφέρουμε ότι η εφαρμογή που υλοποιήσαμε σχεδιάστηκε με πραγματικά δεδομένα και τα έγραφα xls των οποίων αυτοματοποιήσαμε την διαδικασία δημιουργίας τους είναι αληθινά.

Στο κεφάλαιο δύο παρουσιάζονται τα γενικά χαρακτηριστικά συστημάτων διαχείρισης στόλου και οι λειτουργίες τους. Επίσης αναφερόμαστε σε σύγχρονα συστήματα διαχείρισης σε πραγματικό χρόνο, στην δρομολόγηση και στην διαδικασία διαχείρισης του στόλου.

Στο κεφάλαιο τρία και τέσσερα αναφερόμαστε στον σχεδιασμό, την υλοποίηση καθώς και τη περιγραφή και παρουσίαση της εφαρμογής .

Κεφάλαιο 2

Συστήματα Διαχείρισης Στόλου και Δίκτυα Διανομής

2.1 Συστήματα Διαχείρισης

Τα Συστήματα διαχείρισης στόλου οχημάτων παρέχουν στους διαχειριστές του στόλου μια σειρά από ολοκληρωμένες λύσεις διαχείρισης. Στην συνέχεια θα αναλύσουμε τα γενικά χαρακτηριστικά και τις λειτουργίες συστημάτων διαχείρισης. Επίσης θα γίνει αξιολόγηση των υπαρχόντων συστημάτων διαχείρισης στόλου οχημάτων.

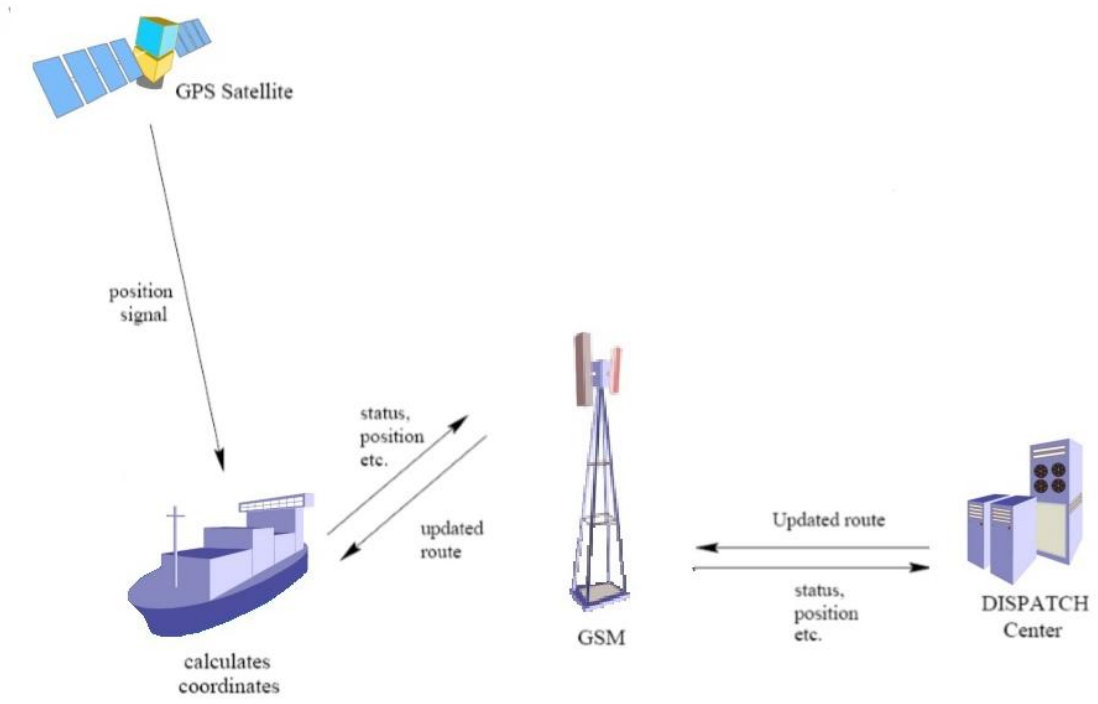
2.1.1 Γενικά χαρακτηριστικά συστημάτων διαχείρισης στόλου

Τα Συστήματα Διαχείρισης Στόλου επιτρέπουν την παρακολούθηση διαφόρων παραμέτρων σε πραγματικό χρόνο, όπως η θέση και η ταχύτητα του οχήματος, τα δεδομένα για το φορτίο (π.χ. η θερμοκρασία το βάρος) και ούτω καθεξής, προκειμένου να εντοπίσει τα σημεία συμφόρησης κατά την εκτέλεση και ως την παράδοση με στόχο την ελαχιστοποίηση των λειτουργικών εξόδων. Τα συστήματα αυτά είναι εξειδικευμένα πακέτα λογισμικού που απευθύνονται ειδικά σε εργασίες στόλου (Laporte και Crainic, 2000)[1]. Η αρχιτεκτονική του συστήματος απεικονίζεται στο σχήμα 2.01 και αποτελείται από τρία κύρια συστατικά:

On-board τηλεματικό εξοπλισμό: Αποτελείται από ένα σύνολο μικροηλεκτρονικών συσκευών που είναι υπεύθυνες για την επεξεργασία όλων των δεδομένων που λαμβάνονται είτε από τους δορυφόρους εντοπισμού θέσης ή από τους αισθητήρες επί του σκάφους και τους διαβιβάζει μέσω ενός δικτύου κινητής επικοινωνίας στο κέντρο ελέγχου.

Το κέντρο ελέγχου: Αποτελείται από μια εφαρμογή που διαχειρίζεται όλα τα δεδομένα που μεταδίδονται από το εποχούμενο υλικό. Ο υπεύθυνος για το σχεδιασμό διαδρομής είναι σε θέση να γνωρίζει σε πραγματικό χρόνο τη θέση κάθε οχήματος. Επίσης μπορεί να συλλέγει όλες τις αναγκαίες πληροφορίες σχετικά με την εκτέλεση του κάθε χρονοδιαγράμματος παράδοσης.

Mobile & Satellite Συστήματα Επικοινωνίας: Αποτελούνται από δύο μέρη: α) Η κινητή πρόσβαση στο επίγειο δίκτυο (π.χ. GSM), η οποία είναι υπεύθυνη για την ασύρματη διασύνδεση του κέντρου ελέγχου με τις συσκευές επί του σκάφους, και β) το σύστημα εντοπισμού θέσης (π.χ. GPS), το οποίο είναι υπεύθυνο για τον εντοπισμό του οχήματος.



Σχήμα 2.01 Αρχιτεκτονική ενός τυπικού συστήματος Διαχείρισης Στόλου (Larsen, 2000)

2.1.2 Λειτουργίες συστημάτων διαχείρισης στόλου

Τα συστήματα διαχείρισης στόλου δίνουν την δυνατότητα στους διαχειριστές του στόλου να παρακολουθούν τον στόλο τους μέσα από ηλεκτρονικούς χάρτες έτσι συντελούν στη βελτίωση της απόδοσης του δικτύου διανομής (Gruhn et al., 2003) [2]. Ιδανικά, το κέντρο διανομής γνωρίζει σε ποιο κράτος είναι το όχημα σε κάθε δεδομένη χρονική στιγμή. Ρυθμίσεις και πληροφορίες θέσης πραγματικού χρόνου μεταδίδονται σε τακτά χρονικά διαστήματα και ένα (interpolation scheme) σχήμα παρεμβολής χρησιμοποιείται προκειμένου να εκτιμήσει τις θέσεις των οχημάτων. Οι κύριες λειτουργίες των εν λόγω συστημάτων είναι (Rushton et al, 2000.) [3]:

Η διαχείριση και η κοστολόγηση του στόλου: Αυτή η δυνατότητα παρέχει λεπτομερείς πληροφορίες σχετικά με το κόστος του οχήματος και του στόλου. Βοηθά τον logistics manager, παρέχοντας αναλύσεις και πληροφορίες σχετικές με το κάθε όχημα ξεχωριστά αλλά και για τη συνολική αποδοτικότητα του στόλου. Τα χαρακτηριστικά του οχήματος περιλαμβάνουν την ανάλυση του κόστους του οδηγού, καθώς και το συνολικό κόστος του στόλου.

Συστήματα Τηλεματικής καταγραφής , ανάλυσης και εκ των υστέρων αναφορές: Οι πληροφορίες μπορούν να παρέχονται μέσω του καταγραφέα Τηλεματικής ηχογράφησης δηλαδή παρέχονται τα δεδομένα εισόδου για την ανάλυση της απόδοσης του οδηγού και του οχήματος. Υπάρχουν διαθέσιμα συστήματα που έχουν την δυνατότητα να λάβουν τα δεδομένα εισόδου και να παράγουν αναφορές εκ των υστέρων για το χρόνο ανάπαυσης, το χρόνο οδήγησης και τον χρόνο διαλλείματος , καθώς και λεπτομέρειες για τις νομικές παραβάσεις.

Προγραμματισμός συντήρησης: Αυτό περιλαμβάνει την παρακολούθηση της ζωής των οχημάτων του στόλου και τον προγραμματισμό τακτικής και μη τακτικής συντήρησης και επισκευής. Κάποια χαρακτηριστικά γνωρίσματα είναι το ιστορικό service, αναφορές προγράμματος συντήρησης και ανάλυση κόστους συνεργείων .

2.1.3 Αξιολόγηση των υπαρχόντων συστημάτων διαχείρισης στόλου οχημάτων

Τα υπάρχοντα συστήματα διαχείρισης στόλου χρησιμοποιούν κινητές τεχνολογίες και τεχνολογίες εντοπισμού θέσης για τον έλεγχο της εκτέλεσης του αρχικού χρονοδιαγράμματος παράδοσης. Τα συστήματα παρακολούθησης χρησιμοποιούνται κυρίως για σκοπούς επιτήρησης του στόλου. Πράγματι, τα συστήματα αυτά είναι σε θέση να συλλέγουν και να επεξεργάζονται, σε πραγματικό χρόνο, διάφορα στοιχεία που διαβιβάζονται από τα οχήματα (π.χ. γεωγραφικές συντεταγμένες, ταχύτητα του οχήματος, θερμοκρασία φορτίου και

ούτω καθεξής), με αυτόν τον τρόπο δημιουργούνται αναφορές που περιλαμβάνουν πληροφορίες σχετικά με την καθημερινή διαδρομή του κάθε οχήματος.

Στη βιβλιογραφία, παρατηρούνται κυρίως δύο περιπτώσεις όπου χρησιμοποιούνται συστήματα οθόνων. Στην πρώτη περίπτωση, τα συστήματα αυτά εφαρμόζονται σε δίκτυα διανομής στην πραγματική ζωή, έτσι ώστε να γίνεται σύγκριση της απόδοσης της εκτέλεσης της παροχής πριν και μετά τη χρήση του συστήματος.

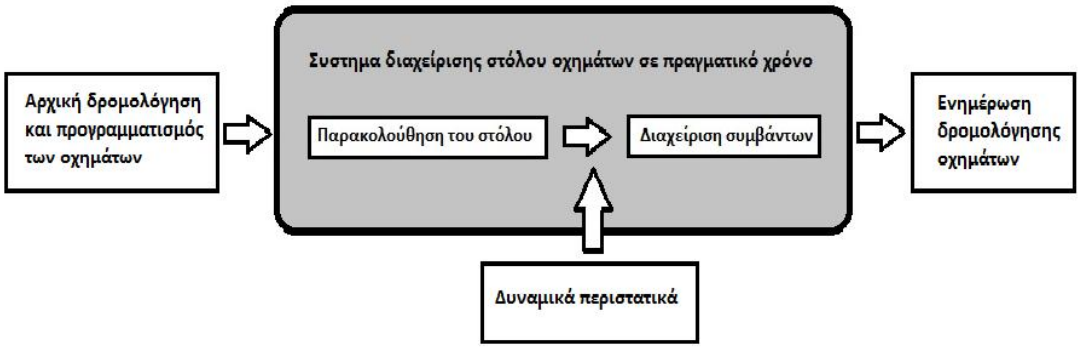
Τα αποτελέσματα της βιβλιογραφίας μας δείχνουν ότι η χρήση των εν λόγω συστημάτων είναι σε θέση να μειώσει το κόστος της διαδικασίας παράδοσης, ακόμα ο σχεδιαστής της διαδρομής είναι σε θέση να εντοπίσει προβληματικούς πελάτες (δηλαδή τους πελάτες με αυξημένους χρόνους εξυπηρέτησης), καθώς και τους δρόμους με μεγάλη κυκλοφοριακή συμφόρηση. Στη δεύτερη περίπτωση, οι ερευνητές χρησιμοποιούν συστήματα παρακολούθησης ώστε να εκτιμηθεί η απόδοση των αρχικών αλγορίθμων δρομολόγησης που έχουν εφαρμοστεί για να κατασκευαστεί η βέλτιστη διαδρομή που το κάθε όχημα θα πρέπει να ακολουθήσει. Έτσι, στη δεύτερη περίπτωση, τα συστήματα παρακολούθησης χρησιμοποιούνται μόνο για τη συλλογή δεδομένων που μπορούν να βοηθήσουν έναν ερευνητή να διερευνήσει κατά πόσον σημαντικά μειώνουν το κόστος λειτουργίας του στόλου οι προτεινόμενοι αλγόριθμοι .

Παρόλο που η χρήση των εν λόγω συστημάτων υποστηρίζει την καλύτερη αξιοποίηση και διαχείριση του στόλου για τις διανομές, δεν είναι συνήθως σχεδιασμένα για την αντιμετώπιση απρόβλεπτων γεγονότων κατά συστηματικό τρόπο. Αυτό οφείλεται στις διαταραχές που είναι εγγενείς, ιδίως στο αστικό περιβάλλον και προκύπτουν από την κυκλοφοριακή συμφόρηση, καιρικές συνθήκες, βλάβες των οχημάτων, και άλλα απρόβλεπτα γεγονότα [4](Psaraftis, 1995, [5] Savelsbergh and Sol, 1998). Το επόμενο τμήμα εξετάζει τα κύρια χαρακτηριστικά των συστημάτων διαχείρισης στόλου οχημάτων σε πραγματικό χρόνο, αυτά μπορούν να αντιμετωπίσουν και να διαχειριστούν απροσδόκητα γεγονότα με τη χρήση πληροφοριών που λαμβάνουν με δυναμικό τρόπο.

2.2 Συστήματα Διαχείρισης Στόλου σε πραγματικό χρόνο

Τα προαναφερθέντα συστήματα χρησιμοποιούνται κυρίως για σκοπούς παρακολούθησης και δεν έχουν σχεδιαστεί για να διαχειρίζονται ένα απροσδόκητο συμβάν κατά την εκτέλεση της αποστολής κάποιου οχήματος. Ωστόσο, τις τελευταίες δεκαετίες, η έρευνα επικεντρώθηκε στη διαχείριση του στόλου σε πραγματικό χρόνο που είναι σε θέση να διαχειριστεί μια σειρά από γεγονότα, όπως τα νέα αιτήματα των πελατών. Συνήθως τέτοια συστήματα προσπαθούν να εκπληρώσουν δύο βασικούς στόχους: Ο πρώτος στόχος είναι η ελαχιστοποίηση των καθυστερήσεων και ο δεύτερος είναι η ελαχιστοποίηση του κόστους, η οποία θεωρείται ότι είναι ανάλογη με τη συνολική διάρκεια όλων των διαδρομών.

Οι βασικές συνιστώσες που ενσωματώνουν τα συστήματα διαχείρισης του στόλου σε πραγματικό χρόνο φαίνεται στο σχήμα 2.02. Όπως μπορεί να δει κανείς σε τέτοια συστήματα, εκτός από τη διαδικασία παρακολούθησης του στόλου, υπάρχει συνήθως ένας μηχανισμός διαχείρισης γεγονότων που χειρίζεται με δυναμικό τρόπο απροσδόκητα γεγονότα όπως η κυκλοφοριακή συμφόρηση, βλάβες οχημάτων όπως και τα νέα αιτήματα των πελατών. Η ανίχνευση των συμβάντων γίνεται συνήθως μέσω των μεθόδων πρόβλεψης του χρόνου του ταξιδιού και αντιμετώπισης περιστατικών (π.χ. αλλαγή πορείας του οχήματος) γίνεται μέσω αλγορίθμων δρομολόγησης.



Σχήμα 2.02 Κύρια συστατικά στοιχεία ενός συστήματος διαχείρισης στόλου οχημάτων πραγματικού χρόνου

2.2.1 Σύγχρονα συστήματα διαχείρισης στόλου οχημάτων σε πραγματικό χρόνο

Τα σύγχρονα συστήματα διαχείρισης στόλου οχημάτων σε πραγματικό χρόνο μπορούν να κατηγοριοποιηθούν ανάλογα με το είδος των πληροφοριών πραγματικού χρόνου που επεξεργάζονται. Ο Πίνακας 2.03 δείχνει μια λίστα των ερευνητικών εργασιών στον τομέα της διαχείρισης του στόλου σε πραγματικό χρόνο.

Authors/Reference	Type of real-time information		
	Client request	Variable travel times	
		Traffic Mgt Centre	On-board systems
Goetschalckx (1988)	X		
Powel (1990)	X		
Savelsbergh & Sol (1997)	X		
Slater (2002)	X		
Gans & Ryzin (1999)	X		
Kim et al. (2003)		X	
Ichoua et al. (2003)			X
Fleischmann et al. (2004b)	X	X	
Taniguchi & Shimamoto (2004)		X	
Hanghani & Jung (2004)	X		X

Πίνακας 2.03 Συστήματα διαχείρισης στόλου σε πραγματικό χρόνο από την βιβλιογραφία.

Στην περίπτωση που προκύψει κάποιο αίτημα από νέο πελάτη πρέπει να τηρείται σε μια συγκεκριμένη χρονική περίοδο. Αρχικά τα συστήματα που εμφανίστηκαν κατά τις προηγούμενες δεκαετίες (Goetschalckx 1998, Powel 1990, Savelsbergh & Sol, 1997 Slater 2002, Gans & Ryzing 1999) αν και μπορούν να ανταπεξέλθουν στις νέες απαιτήσεις των πελατών υποθέτουν ότι οι χρόνοι ταξιδίου στη διάρκεια της ημέρας είναι σταθεροί ή εκμεταλλεύονται απλές διαδικασίες για την προσαρμογή τους. Εντούτοις, οι παραδοχές αυτές είναι αδύνατον να προσεγγίσουν τις συνθήκες του πραγματικού κόσμου όπου οι χρόνοι ταξιδιού υπόκεινται σε πιο ανεπαίσθητες διακυμάνσεις στην πάροδο του χρόνου. Οι μεταβολές αυτές μπορούν να προκύψουν από διάφορα προβλέψιμα γεγονότα (π.χ. συμφόρηση κατά τις ώρες αιχμής) ή από απρόβλεπτα γεγονότα, όπως ατυχήματα, μηχανικές βλάβες και άλλα. Ως εκ τούτου, ιδανική λύση σε μια σύνθεση μιας αστικής διακίνησης

εμπορευμάτων, προϋποθέτει σταθερό χρόνο μετακίνησης αυτό όμως μπορεί να μην είναι ιδανικό η να είναι ακόμα και ανέφικτό [6] (Ichoua., 2003).

Ο Kim (2003) [7] ήταν ο πρώτος που εισήγαγε πληροφορίες για την κυκλοφορία σε πραγματικό χρόνο, σε ένα τέτοιο σύστημα. Εξέτασε την αξία της βέλτιστης δρομολόγησης των οχημάτων σε μη στατικό στοχαστικό δίκτυο με την ανάπτυξη ενός συστήματος προσέγγισης για τον προσδιορισμό του χρόνου εργασίας του οδηγού, την βέλτιστη ώρα αναχώρησης και τις βέλτιστες πολιτικές δρομολόγησης σύμφωνα με στοχαστικά μεταβαλλόμενη ροή της κυκλοφορίας. Ο Ichoua (2003), παρουσίασε ένα μοντέλο διαχείρισης στόλου σε πραγματικό χρόνο με βάση το χρόνο και εξαρτάται από τις ταχύτητες του ταξιδιού. Μια πειραματική αξιολόγηση του προτεινόμενου μοντέλου γίνεται σε ένα στατικό και ένα δυναμικό περιβάλλον. Τα αποτελέσματα μας δείχνουν ότι ο χρόνος που εξαρτάται από το μοντέλο παρέχει σημαντικές βελτιώσεις σε σχέση με ένα μοντέλο που βασίζεται σε σταθερές ώρες ταξιδιών. Ο Fleischmann (2004)[8], παρουσίασε ένα δυναμικό σύστημα δρομολόγησης που αποστέλλει τον στόλο των οχημάτων σύμφωνα με τις παραγγελίες των πελατών που φθάνουν τυχαία κατά τη διάρκεια της περιόδου προγραμματισμού. Το σύστημα διαθέτει πληροφορίες σε απευθείας σύνδεση κατά την διάρκεια των ταξιδιών από ένα κέντρο διαχείρισης της κυκλοφορίας. Ο Taniguchi και Shimamoto (2004)[9] παρουσίασαν επίσης ένα έξυπνο σύστημα μεταφοράς, με βάση την δυναμική δρομολόγηση οχήματος και τον προγραμματισμό ταξιδιού με μεταβλητούς χρόνους. Τα αποτελέσματα έδειξαν ότι το συνολικό κόστος μειώθηκε κατά την εφαρμογή της δυναμικής δρομολόγησης οχήματος. Επίσης το μοντέλο χρονοδρομολόγισης με πληροφορίες πραγματικού χρόνου βασίζεται σε μεταβλητούς χρόνους ταξιδιού σε σύγκριση με εκείνων των προβλέψεων του μοντέλου (δηλαδή ο τελευταίος παρέχει προβλεπόμενες ώρες ταξίδια με βάση τα ιστορικά δεδομένα). Τέλος, ο Hanghani & Jung (2004)[10] παρουσίασε μια συστημική προσέγγιση για την αντιμετώπιση του προβλήματος δυναμικής δρομολόγησης οχημάτων με χρονικά εξαρτώμενους χρόνους ταξιδιού. Επίσης παρουσίαζαν ένα γενετικό αλγόριθμο για την επίλυση του προβλήματος. Ωστόσο, τα συμβάντα περιλαμβάνουν την χρονική διάρκεια του ταξιδιού,

καθυστερήσεις των υπηρεσιών και τις κατανομές των οχημάτων που δεν έχουν αντιμετωπιστεί από οποιοδήποτε από τα εν λόγω συστήματα. Αυτό το ερευνητικό κενό καθιστά την ανάγκη για χειρισμό σε πραγματικό χρόνο των διάφορων συμβάντων.

2.2.2 Ανάγκη χειρισμού συμβάντων σε πραγματικό χρόνο

Υπάρχουν τρεις κύριες πηγές των συμβάντων στις παραδόσεις εμπορευματικών μεταφορών: α) συμβάντα που προέρχονται από τους πελάτες που εξυπηρετούνται, β) συμβάντα της οδικής υποδομής και του περιβάλλοντος και γ) συμβάντα που προκύπτουν από τα οχήματα παράδοσης. Οι τρέχουσες έρευνες έχουν ασχοληθεί με μια συγκεκριμένη περίπτωση χειρισμού των συμβάντων (δηλ. οι αιτήσεις των πελατών) και συνεπώς, εξακολουθεί να υπάρχει έλλειψη προσέγγισεις των μοντέλων που προσπαθούν να αντιπροσωπεύσουν πραγματικές συνθήκες, όπως βλάβες του οχήματος, χρόνος ταξιδιού και χρονικές υστερήσεις υπηρεσιών που προκαλούνται από την κυκλοφοριακή συμφόρηση, τιε καιρικές συνθήκες, ή τις εγκαταστάσεις εκφόρτωσης στο χώρο του πελάτη.

2.3 Δίκτυα Διανομής

Ένα μεγάλο τμήμα της επιστήμης της Επιχειρησιακής Έρευνας ασχολείται με προβλήματα δικτύων διανομής. Τα τελευταία χρόνια αναγνωρίζεται η μεγάλη σημασία βέλτιστου σχεδιασμού δικτύων διανομής στοχεύοντας στη μείωση του κόστους αλλά και την καλύτερη εξυπηρέτηση των πελατών. Συνηθέστερα προβλήματα που σχετίζονται με τα δίκτυα διανομής, είναι η διανομή προϊόντων από τα σημεία παραγωγής ή αποθήκευσής τους στα σημεία κατανάλωσής τους, η μεταφορά προσωπικού από και προς το χώρο εργασίας τους, η συλλογή σκουπιδιών, η μεταφορά αλληλογραφίας, η δρομολόγηση πλοίων και αεροπλάνων.

Η γενικότερη και απλούστερη μορφή τέτοιων προβλημάτων αναφέρεται στη σχεδίαση ενός δικτύου διαδρομών μεταξύ κάποιων δεδομένων κόμβων-πελατών που βρίσκονται σε γνωστές γεωγραφικά διεσπαρμένες τοποθεσίες και ενός κόμβου-αποθήκης σε γνωστή θέση, κατά τέτοιο τρόπο, ώστε να δημιουργηθούν κάποια δρομολόγια οχημάτων. Τα δρομολόγια αυτά θα πρέπει να ικανοποιούν την ζήτηση του κάθε κόμβου. Υπάρχουν όμως και επιπλέον περιορισμοί, που αναφέρονται στις χωρητικότητες, τον αριθμό των οχημάτων, στη χρονική διάρκεια, το μήκος των διαδρομών, τις ώρες εργασίας, τα περιθώρια εξυπηρέτησης αλλά και άλλες ειδικότερες προϋποθέσεις που τίθενται από το εκάστοτε επιχειρησιακό μοντέλο. Σε κάθε πρόβλημα αντιστοιχεί κάποια αντικειμενική συνάρτηση που θα πρέπει να βελτιστοποιηθεί. Το συνηθέστερο κριτήριο βελτιστοποίησης σχετίζεται με το συνολικό χιλιομετρικό μήκος που διανύουν τα οχήματα στο δίκτυο.

Τα προβλήματα αυτά, διατηρούν αμείωτο το ερευνητικό ενδιαφέρον κατά τη διάρκεια των τελευταίων χρόνων. Τεράστια πρόοδος έχει γίνει σε αυτόν τον τομέα, τόσο από θεωρητικής, όσο και εφαρμοσμένης πλευράς. Καινούργιες μεθοδολογίες, καθώς και αποτελεσματικές τεχνικές επίλυσης έχουν κατά καιρούς προταθεί, ενώ ταυτόχρονα έχει καταγραφεί μια πολύ μεγάλη λίστα από πρακτικές εφαρμογές. Επιπλέον παρουσιάζεται μεγάλο πρακτικό ενδιαφέρον, αφού οι επιχειρήσεις προκειμένου να επιβιώσουν στη ραγδαία εξελισσόμενη αγορά, τροποποιούν τις εσωτερικές διαδικασίες παραγωγής, εφοδιασμού, αποθήκευσης και διανομής των προσφερομένων αγαθών ή υπηρεσιών τους, δημιουργώντας την ανάγκη για συστήματα που θα υποστηρίξουν αυτές τις νέες λειτουργίες. Η σύγχρονη τάση της αγοράς που επιβάλει συγχωνεύσεις και εξαγορές συγκεντρώνει τα προβλήματα διανομής σε λιγότερους αλλά και μεγαλύτερους φορείς, αυξάνοντας εκθετικά την πολυπλοκότητα του προβλήματος, και δημιουργώντας νέες προκλήσεις για την επιστημονική κοινότητα.

Η ύπαρξη σύγχρονων πληροφοριακών συστημάτων δίνει την δυνατότητα στις εταιρείες που τα χρησιμοποιούν να γνωρίζουν ανά πάσα στιγμή τις πληροφορίες που επιθυμούν για τους πελάτες και τις ιδιαιτερότητες στην εξυπηρέτησή τους, τις παραγγελίες, τα διακινούμενα προϊόντα, τα οχήματα διανομής. Παράλληλα,

παρέχεται η δυνατότητα με την υποβοήθηση κατάλληλων ηλεκτρονικών συσκευών να εντοπίζεται η ακριβής θέση κάθε οχήματος, των συνθηκών μεταφοράς, καθώς επίσης και δυνατότητα άμεσης επικοινωνίας με τα οχήματά σε περιπτώσεις ανάκλησης μίας παραγγελίας που ακυρώθηκε, προσθήκης έκτακτης παραγγελίας, αναπροσαρμογής του δρομολογίου για διαφόρους λόγους. Η αξιοποίηση όλων αυτών των πληροφοριών με στόχο την βελτιστοποίηση του δικτύου, θα έχει σαν αποτέλεσμα καλύτερο επίπεδο εξυπηρέτησης πελατών, μεγαλύτερη ευελιξία και σημαντικές μειώσεις στο κόστος διανομής.

2.3.1 Δρομολόγηση Στόλου Οχημάτων σε Δίκτυα Διανομής

Η έρευνα πάνω στην δρομολόγηση στόλου οχημάτων πραγματοποιήθηκε στα πλαίσια του ευρωπαϊκού ερευνητικού DYNALOG προγράμματος [11] για την διευθέτηση ενός σύνθετου Προβλήματος Δρομολόγησης Στόλου Οχημάτων (ΠΔΣΟ) με Παράθυρα Χρόνου (ΠΔΣΟΠΧ) και στοχαστική ζήτηση σε δίκτυα διανομής. Το ΠΔΣΟΔΔ συνεπάγεται την δρομολόγηση ενός στόλου οχημάτων, πεπερασμένης χωρητικότητας και χρόνου κίνησης, από μια (ή περισσότερες) αποθήκες-αφετηρίες σε ένα σύνολο διασκορπισμένων πελατών με στοχαστική ζήτηση μέσα σε προκαθορισμένα παράθυρα χρόνου. Η ζήτηση κάθε πελάτη είναι στοχαστική. Ο στόχος είναι η δημιουργία ενός αριθμού βασικών διαδρομών για μια συγκεκριμένη χρονική περίοδο που προσδιορίζεται από το χρήστη έτσι ώστε να ικανοποιούνται ταυτόχρονα και κάποια συγκεκριμένα πρόσθετα κριτήρια όπως οι ώρες εργασίας των οδηγών, το ελάχιστό κέρδος για κάθε διαδρομή, η προκαθορισμένη (α)συμβατότητα μεταξύ οδηγών και πελατών.

Η έρευνα πάνω στο θέμα είναι εκτεταμένη, ωστόσο λόγω της πολυπλοκότητας του προβλήματος υπάρχει χώρος για επιπρόσθετη έρευνα για την εύρεση καλύτερων και υπολογιστικά γρηγορότερων προσεγγίσεων της βέλτιστης λύσης. Αξίζει να αναφερθεί ότι ακόμα και η εύρεση μιας δυνατής και όχι βέλτιστης λύσης για το πρόβλημα είναι αδύνατη σε πολυωνυμικό χρόνο, το πρόβλημα είναι NP δύσκολο (Savelsberg [12]).

Γενικά η χρήση γενετικών αλγορίθμων (ΓΑ) για την αντιμετώπιση του προβλήματος είναι πολύ ελπιδοφόρα. Σε αντίθεση με τις κλασικές μεθόδους αναζήτησης, οι ΓΑ δεν εξερευνούν το χώρο λύσεων διαδοχικά με εξαντλητικό τρόπο, αλλά ένα σύνολο από τρέχουσες λύσεις βελτιώνεται μέσω της στοχαστικής επιλογής μίας ή περισσοτέρων λύσεων και τον συνδυασμό τους για την παραγωγή μίας η περισσοτέρων νέων λύσεων με την εφαρμογή γενετικών τελεστών. Κάθε λύση κωδικοποιείται σε μια δομή δεδομένων που στα πλαίσια ενός ΓΑ ονομάζεται χρωματόσωμα. Βασικά, ένας ΓΑ είναι μια μαθηματική τεχνική αναζήτησης που βασίζεται στις αρχές της φυσικής επιλογής και γενετικής αναπαραγωγής[13]. Ειδικότερα, ένας ΓΑ αρχικά κατασκευάζει ένα σύνολο λύσεων (πληθυσμός του γενετικού αλγόριθμου) και σε κάθε βήμα (γενιά) εφαρμόζει πάνω στις τρέχουσες λύσεις (χρωματοσώματα) γενετικές πράξεις (μετάλλαξη, διασταύρωση) με σκοπό την εύρεση σχεδόν βέλτιστων λύσεων για το πρόβλημα. Στην αρχή κάθε βήματος επιλέγονται οι λύσεις προς εξέλιξη, πραγματοποιείται η αναπαραγωγή (διασταύρωση και μετάλλαξη) και στο τέλος πραγματοποιείται η αντικατάσταση των παλιών λύσεων με τις νέες και αξιολογείται η ποιότητα των λύσεων στο πληθυσμό (υπολογισμός αντικειμενικής συνάρτησης για κάθε λύση). Στη σειριακή εκδοχή του αλγορίθμου η αξιολόγηση των λύσεων στο τέλος κάθε γενιάς πραγματοποιείται σειριακά. Στην παράλληλη εκδοχή μπορούμε να μοιράσουμε στις διαθέσιμες διεργασίες (που εκτελούνται παράλληλα) είτε την αξιολόγηση ολόκληρων λύσεων του πληθυσμού είτε την αξιολόγηση, ανά όχημα, κάθε λύσεως του πληθυσμό. Η εφαρμογή της παραλληλοποίησης κατά την διάρκεια αξιολόγησης του συνολικού πληθυσμού (μοίρασμα ολόκληρων λύσεων στις διαθέσιμες διεργασίες) έδωσε σημαντικά χαμηλότερους χρόνους εκτέλεσης σε σύγκριση με την σειριακή εκδοχή και οι διεργασίες έδειξαν σχεδόν το ίδιο ποσοστό χρησιμοποίησης και σχεδόν τον ίδιο χρόνο εκτέλεσης. Το σύστημα που περιλαμβάνει τον ΓΑ υποστηρίζει όλη την απαραίτητη λειτουργικότητα για την υποστήριξη της λήψης αποφάσεων σχετικά με τον στρατηγικό σχεδιασμό των δρομολογίων. Επιτρέπει στον χρήστη να επιλέγει τους πελάτες και τα φορτηγά που θα λάβουν μέρος στην διαδικασία δρομολόγησης, να καθορίζει τις παραμέτρους αναζήτησης και να διαχειρίζεται τα δρομολόγια που έχουν υπολογιστεί. Αξίζει να αναφερθεί ότι το

ΠΔΣΟΔΔ, εκτός από την διανομή προϊόντων όπως στην περίπτωση μας, μπορεί να εφαρμοστεί στην διανομή ταχυδρομείου ή καυσίμων, παραλαβή, μεταφορά και παράδοση αγαθών και την συλλογή απορριμμάτων.

2.4 Διαδικασία Διαχείρισης Στόλου πλοίων

Η διαδικασία διαχείρισης του στόλου εξαρτάται από την ναυτιλιακή αγορά. Ο διαχωρισμός όμως της ναυτιλιακής αγοράς σε επιμέρους ανεξάρτητες αγορές, η ύπαρξη διαφορετικών εμπλεκόμενων ρόλων με αντικρουόμενα συμφέροντα και το πλήθος των διαφορετικών λειτουργιών που πρέπει να διεκπεραιωθούν προκειμένου να εκτελεστεί ένα ταξίδι μεταφοράς φορτίου καθιστά αδύνατη την ενασχόληση μιας ερευνητικής εργασίας με το σύνολο της ναυτιλιακής αγοράς. Για τον λόγο αυτό θα επικεντρωθούμε σε συγκεκριμένα τμήματα της ναυτιλιακής αγοράς που διαδραματίζουν καταλυτικό παράγοντα για την διαχείριση στόλου πλοίων και τις αντίστοιχες διαδικασίες. Ποιο συγκεκριμένα θα αναφερθούμε στα εξής:

- Η επιλογή φορτίων, η οποία είναι μια διεργασία που εκτελείται από τους πλοιοκτήτες, εφοπλιστές, πράκτορες ναυτιλιακών εταιρειών για πλοία που λειτουργούν στην ελεύθερη αγορά.
- Η δρομολόγηση ενός στόλου μιας εταιρείας προκειμένου να μεταφερθεί ένα σύνολο φορτίων με το βέλτιστο δυνατό τρόπο. Η διεργασία αυτή εκτελείται από τους πλοιοκτήτες και τους εφοπλιστές, που δραστηριοποιούνται κυρίως στην βιομηχανική ναυτιλία.

2.4.1 Διαδικασία επιλογής φορτίων

Η διαδικασία αυτή εκτελείται για τα πλοία που δραστηριοποιούνται στην ελεύθερη αγορά καθώς στην περίπτωση της βιομηχανικής ναυτιλίας η ναυτιλιακή εταιρεία δεν αναζητά φορτία αλλά μεταφέρει όλα τα φορτία που της ζητείται να μεταφέρει

για λογαριασμό της βιομηχανίας στην οποία ανήκει. Ο σκοπός της διαδικασίας αυτής είναι να βρεθούν εκείνα τα φορτία της αγοράς τα οποία αναμένεται να είναι τα πιο κερδοφόρα Fagerholt, Christiansen (2000)[14]. Τα κριτήρια που χρησιμοποιούνται για την αξιολόγηση των φορτίων ποικίλουν ανάλογα με την πολιτική της κάθε ναυτιλιακής εταιρείας.

Η διαδικασία όμως που ακολουθούν οι ναυτιλιακές εταιρείες ή οι μεσίτες τους, για να επιλέξουν ένα σύνολο από φορτία τα οποία θα αξιολογηθούν στην συνέχεια με μεγαλύτερη ακρίβεια, είναι σταθερή ανεξάρτητη από την πολιτική της εταιρείας. Την διαδικασία αυτή την εκτελούν στελέχη με μεγάλη εμπειρία στην ναυλαγορά τα οποία έχουν την ικανότητα να αναγνωρίσουν τις ευκαιρίες και τους κινδύνους που κρύβονται στην αγορά μέσα από ένα μεγάλο σύνολο δεδομένων και παραμέτρων που πρέπει να εξεταστούν. Η διαδικασία αυτή ξεκινάει βασικά από τον ορισμό του πλοίου για το οποίο αναζητείται φορτίο. Τα στοιχεία του πλοίου που απαιτούνται να είναι γνωστά έτσι ώστε να ξεκινήσει η διαδικασία επιλογής φορτίων είναι τα εξής:

1) Η θέση του πλοίου και η εκτιμώμενη ημερομηνία που αυτό προβλέπεται να είναι διαθέσιμο. Τα στοιχεία αυτά προκύπτουν από το τελευταίο προγραμματισμένο ταξίδι του πλοίου.

2) Ο τύπος του πλοίου

3) Ο αριθμός των αμπαριών του πλοίου και η χωρητικότητα του κάθε ενός από αυτά. Στην περίπτωση που πρόκειται για πλοία τύπου Tweendecker απαιτείται η γνώση της χωρητικότητας και τις διάταξης των διαμερισμάτων που δημιουργούνται αν τοποθετηθούν διαχωριστικά σε κάθε επιτρεπτή θέση. Στην ίδια περίπτωση είναι απαραίτητο να είναι γνωστό και το πλήθος των διαχωριστικών που είναι διαθέσιμα.

4) Το μήκος και το βύθισμα του πλοίου

5) Οι ταχύτητες του πλοίου όταν αυτό είναι με ή χωρίς φορτίο αντίστοιχα

6) Σε περίπτωση που υπάρχουν συμφωνίες για μεταφορά φορτίων σε μεταγενέστερο χρονικό διάστημα από το διάστημα για το οποίο αναζητείται φορτίο, τότε πρέπει να είναι γνωστά όλα τα στοιχεία και οι δεσμεύσεις που έχουν γίνει για τις συμφωνηθείσες μεταφορές.

7) Χρονικά διαστήματα κατά τα οποία το πλοίο πρέπει να παραμείνει ανενεργό και το λιμάνι στο οποίο πρέπει να παρευρίσκεται.

Στην συνέχεια η εταιρεία ελέγχει τα φορτία για τα οποία έχει στοιχεία. Τα απαραίτητα στοιχεία, πέρα από αυτά της ιδιοκτησίας και της αντιπροσώπευσης, που πρέπει να είναι διαθέσιμα για ένα φορτίο για να ληφθεί αυτό υπόψη στην διαδικασία της επιλογής φορτίων είναι :

1) Το είδος του φορτίου

2) Η ποσότητα του φορτίου

3) ο συντελεστής στοιβασίας του φορτίου, απαραίτητος για την μετατροπή της ποσότητας του φορτίου σε όγκο φορτίου

4) το λιμάνι φόρτωσης

5) το διάστημα που αυτό θα είναι διαθέσιμο στο λιμάνι φόρτωσης

6) το λιμάνι εκφόρτωσης

7) αν υπάρχει η ημερομηνία νωρίτερα από την οποία δεν μπορεί να ξεκινήσει η εκφόρτωση του φορτίου

8) αν υπάρχει η καταληκτική ημερομηνία παράδοσης φορτίου

9) οι ρυθμοί φόρτωσης και εκφόρτωσης

10) η τρέχουσα τιμή ναύλου που επικρατεί στην αγορά για το συγκεκριμένο είδος φορτίου και για τον εμπορικό δρόμο που ανήκει η διαδρομή μεταξύ λιμανιών φόρτωσης και εκφόρτωσης του συγκεκριμένου φορτίου.

Την πληροφορία για τα διαθέσιμα φορτία η εταιρεία την λαμβάνει από το δίκτυο της ενημέρωσης το οποίο διαθέτει στο οποίο συμμετέχουν ναυλωτές, μεσίτες και πράκτορες με τους οποίους η εταιρεία συνεργάζεται και θεωρεί αξιόπιστη την πληροφόρηση τους. Ο τρόπος με τον οποίο λαμβάνει την πληροφορία η εταιρεία δεν είναι τυποποιημένος ούτε ως προς το μέσον ούτε ως προς την μορφοποίηση. Έτσι η πληροφορία για τα διαθέσιμα φορτία είναι άλλοτε σε έντυπη μορφή (fax), άλλοτε σε ηλεκτρονική μορφή (email) και άλλοτε η πληροφόρηση είναι μόνο τηλεφωνική.

Με δεδομένα όλα τα παραπάνω στοιχεία και θεωρώντας γνωστές τις αποστάσεις μεταξύ των όλων των εμπορικών λιμανιών, το τμήμα ναύλωσης βρίσκει από το αρχικό σύνολο των διαθέσιμων φορτίων εκείνο το υποσύνολο που περιέχει εκείνα τα φορτία που μπορούν να μεταφερθούν από το ενδιαφερόμενο πλοίο. Ένα φορτίο θεωρείται ότι μπορεί να μεταφερθεί από ένα πλοίο όταν:

1) Το είδος του φορτίου μπορεί να μεταφερθεί από τον τύπο πλοίων που ανήκει το ενδιαφερόμενο πλοίο

2) Το μέγεθος του φορτίου τόσο σε ποσότητα όσο και σε όγκο είναι μικρότερο από την διαθέσιμη χωρητικότητα του πλοίου

3) Το μέγεθός του πλοίου επιτρέπει την προσέγγιση των λιμανιών φόρτωσης και εκφόρτωσης

4) Το πλοίο προλαβαίνει να μεταβεί από το λιμάνι στο οποίο θα βρίσκεται διαθέσιμο στο λιμάνι εκφόρτωσης και να φορτώσει το φορτίο πριν επέλθει η ημερομηνία λήξης φόρτωσης.

5) Σε περίπτωση που υπάρχει καταληκτική ημερομηνία παράδοσης του πλοίου να προλαβαίνει το πλοίο να ξεφορτώσει το φορτίο πριν επέλθει η ημερομηνία αυτή.

6) Στην περίπτωση κατά την οποία το πλοίο είναι ήδη φορτωμένο με κάποιο φορτίο και αναζητείται νέο φορτίο για παράλληλη μεταφορά θα πρέπει η τοποθέτηση του στα διαμερίσματα του πλοίου, μαζί με το προηγούμενο

φορτίο, να είναι εφικτή. Αν η φόρτωση είναι εφικτή καθορίζεται από την διαθέσιμη χωρητικότητα που προκύπτει με βάση την τοποθέτηση των ήδη υπαρχόντων φορτίων στο πλοίο και από το πλήθος των διαμερισμάτων που διαθέτει το πλοίο το οποίο δεν μπορεί να είναι μικρότερο από το πλήθος των φορτίων που είναι φορτωμένα.

7) Στην περίπτωση που υπάρχουν συμβόλαια για μεταφορά άλλων φορτίων σε μετέπειτα χρονική στιγμή από αυτή που αντιστοιχεί η μεταφορά του υπό εξέταση φορτίου, τότε η μεταφορά του φορτίου που εξετάζεται δεν πρέπει να επηρεάζει την εκτέλεση των ήδη συμφωνηθέντων μεταφορών.

8) Για το χρονικό διάστημα που διαρκεί η μεταφορά του φορτίου το πλοίο να είναι ενεργό και να μην πρέπει να παρευρεθεί σε κάποιο λιμάνι ανενεργό.

Από το σύνολο των φορτίων που καλύπτουν τους παραπάνω περιορισμούς το τμήμα ναύλωσης πρέπει να επιλέξει εκείνα τα φορτία για τα οποία θα κάνει περαιτέρω αξιολόγηση και για τα οποία ενδεχομένως να προχωρήσει σε διαπραγματεύσεις.

2.4.2 Διαδικασία δρομολόγησης στόλου για την μεταφορά φορτίων

Η διαδικασία δρομολόγησης στόλου για την μεταφορά φορτίων, αφορά την διαδικασία κατά την οποία μια ναυτιλιακή εταιρεία αποφασίζει την ανάθεση προς μεταφορά ενός συνόλου φορτίων στα πλοία που απαρτίζουν το στόλο της και την σειρά φόρτωσης και εκφόρτωσης των φορτίων για κάθε στόλο Geir BrØnmo(2007) [15]. Η διαδικασία αυτή ακολουθείται τόσο από τις εταιρείες της ελεύθερης ναυτιλίας όσο και από τις εταιρείες της βιομηχανικής ναυτιλίας. Ωστόσο στην περίπτωση της ελεύθερης ναυτιλίας η διαδικασία αυτή για το πια φορτία θα μεταφέρει ένα πλοίο πραγματοποιείται παράλληλα με την διαδικασία επιλογής φορτίων αφού στις συμφωνίες που υπάρχουν με τους ιδιοκτήτες των φορτίων

προσδιορίζεται και το πλοίο που θα μεταφέρει το εκάστοτε φορτίο. Αντίθετα στην περίπτωση της βιομηχανικής ναυτιλίας όπου η μεταφορά των φορτίων γίνεται για λογαριασμό της ίδιας της εταιρείας και όχι τρίτων, δεν υπάρχει καμία αρχική δέσμευση ως προς το πιο πλοίο θα μεταφέρει κάποιο φορτίο. Έτσι στις εταιρείες αυτές απαιτείται να ακολουθηθεί μια διαδικασία σχεδίασης της μεταφοράς των φορτίων. Στις ναυτιλιακές αυτές εταιρείες παραδίδεται η απαιτούμενη ζήτηση για μεταφορά φορτίων για ένα χρονικό διάστημα και η ναυτιλιακή εταιρεία πρέπει να καλύψει την ζήτηση αυτή είτε όσο πιο γρήγορα γίνεται είτε με το μικρότερο δυνατό κόστος, χρησιμοποιώντας τον ιδιόκτητο στόλο της εταιρείας. Σημαντικό χαρακτηριστικό της διαδικασίας αυτής είναι ότι η ζήτηση που παραδίδεται και πρέπει να καλυφθεί από τον στόλο της εταιρείας, θεωρείται σταθερή για το χρονικό διάστημα στο οποίο αναφέρεται. Αυτό συμβαίνει γιατί τα φορτία της ζήτησης αφορούν υλικά που προορίζονται για την λειτουργία μιας καλά οργανωμένης βιομηχανίας, η οποία μπορεί με ακρίβεια να προβλέψει τις ανάγκες της, πολύ πριν χρειαστεί τα υλικά.

Για την εκτέλεση της διαδικασίας δρομολόγησης στόλου για την μεταφοράς των φορτίων, απαιτούνται να είναι γνωστά τόσο για τα πλοία όσο και για τα φορτία, τα ίδια στοιχεία που παρουσιάστηκαν και στην διαδικασία επιλογής φορτίων από τις εταιρείες της ελεύθερης ναυτιλίας στην ενότητα 2.2 Επιπλέον για την ανάθεση ενός φορτίου σε ένα πλοίο πρέπει να ικανοποιούνται όλοι οι περιορισμοί που επίσης παρουσιάζονται στην ενότητα 2.2 Με δεδομένα τα παραπάνω στοιχεία και την ικανοποίηση των περιορισμών, στόχος της διαδικασίας είναι η βέλτιστη ανάθεση των φορτίων στα πλοία έτσι ώστε να μειωθεί είτε το κόστος της μεταφοράς είτε ο χρόνος μεταφοράς των φορτίων. Στην περίπτωση που το πλήθος των φορτίων και των πλοίων είναι μικρό, η βέλτιστη ανάθεση των φορτίων στα πλοία συνήθως προκύπτει άμεσα και η διαδικασία εκτελείται με απλούς υπολογισμούς από έμπειρα στελέχη των εταιρειών. Όσο όμως το πλήθος των φορτίων και των πλοίων αυξάνεται τόσο πιο επιτακτική γίνεται η ανάγκη χρήσης εξειδικευμένων ηλεκτρονικών εργαλείων με την χρήση των οποίων μπορεί να προκύψει η βέλτιστη ανάθεση των φορτίων στα πλοία.

2.4.3 Ανάλυση Ναυτιλιακών Πληροφοριακών Συστημάτων

Γενικά τα προϊόντα τα οποία έχουν κάνει την εμφάνιση τους στην αγορά της ναυτιλίας μπορούν να χωριστούν σε Αυτόνομα Συστήματα (Standalone Systems), Πηγές Άμεσης Πληροφόρησης (Online Information Sources), Περιοχές Ηλεκτρονικού Εμπορίου (Electronic Market Places) και Υβριδικά Συστήματα (Hybrid Systems). Τα αυτόνομα συστήματα όπως τα [16], [17] χρησιμοποιούνται συνήθως για υπολογισμό ταξιδιού (voyage estimation) και ανάλυση χρονοναύλωσης (time charter) και δεν έχουν κάποια ιδιαίτερη ευφυΐα. Οι Πηγές Άμεσης Πληροφόρησης όπως τα [18], [19], [20], [21] παρέχουν στους συνδρομητές πολλές "χρονικές" πληροφορίες όπως δείκτες, θέσεις πλοίων, τιμές καυσίμων κτλ. Οι πληροφορίες που παρέχονται είναι πάρα πολλές και για να εκτιμηθούν χρειάζεται μεγάλη εμπειρία.

Πρέπει να τονιστεί ότι ο όγκος των δεδομένων που έχουν να διαχειριστούν οι ναυτιλιακές εταιρείες, η πολυπλοκότητα των υπολογισμών και η διασπορά της πληροφορίας σε διάφορες πηγές, καθιστά αναγκαία την χρήση πληροφοριακών συστημάτων και του διαδικτύου για την διευκόλυνση της διαδικασίας εύρεσης φορτίου και διαπραγμάτευσης ναύλωσης. Επομένως τα πληροφοριακά συστήματα στην ναυτιλιακή αγορά μπορούν επιπρόσθετα να διαχωριστούν στις εξής κατηγορίες.

Συστήματα εύρεσης και διαχείρισης πληροφορίας: Σε αυτή την κατηγορία ανήκουν τα πληροφοριακά συστήματα που χρησιμοποιούν οι εταιρείες για να βρίσκουν τις αποστάσεις μεταξύ των λιμανιών και τα συστήματα που χρησιμοποιούνται για την μορφοποίηση της πληροφορίας των διαθέσιμων φορτίων προς μεταφορά. Η μη υιοθέτηση μιας διεθνώς αναγνωρισμένης μορφοποίησης της πληροφορίας για την περιγραφή των διαθέσιμων προς μεταφορά φορτίων δυσχεραίνει την χρησιμοποίηση πληροφοριακών συστημάτων που διαχειρίζονται όλα τα διαθέσιμα φορτία για τα οποία είναι ενήμερη μια εταιρεία. Το πρόβλημα αυτό προσπαθούν να το αντιμετωπίσουν κάποια εργαλεία τα οποία εκμεταλλεύονται την χρήση συγκεκριμένων λέξεων και μοτίβων στην ήδη υπάρχουσα πληροφορία για να την

μετατρέψουν σε μια κοινή μορφοποίηση την οποία διαχειρίζονται τα υπόλοιπα πληροφοριακά συστήματα της εταιρείας.

Συστήματα ανάλυσης κόστους ταξιδιού: Με τα συστήματα αυτά, τα οποία χρησιμοποιούνται ευρέως στις ναυτιλιακές εταιρείες, πραγματοποιείται λεπτομερής ανάλυση του κόστους εκτέλεσης ενός συγκεκριμένου ταξιδιού. Στα συστήματα αυτά, ο χρήστης καθορίζει τα στοιχεία ενός φορτίου και τα στοιχεία του πλοίου για το οποίο θέλει να εξετάσει την ναύλωση και το σύστημα πραγματοποιεί λεπτομερή ανάλυση του εκτιμώμενου κόστος της μεταφοράς. Τέτοια συστήματα τις περισσότερες φορές επικοινωνούν με συστήματα υπολογισμού αποστάσεων μεταξύ λιμανιών έτσι ώστε να καθορίσουν το κόστος εκτέλεσης. Τα συστήματα αυτά δίνουν την δυνατότητα στο χρήστη να καθορίσει την τιμή του ναύλου ανά ημέρα ή μεταφερόμενο τόνο, ανάλογα με το τι αυτός επιθυμεί για να υπολογίσουν στην συνέχεια το αναμενόμενο κέρδος. Τέτοιου είδους συστήματα είναι χρήσιμα τόσο για την αξιολόγηση ενός φορτίου όσο και κατά την διαπραγμάτευση της μεταφοράς του.

Συστήματα βέλτιστης ανάθεσης φορτίων σε πλοία: Τα συστήματα αυτά πραγματοποιούν την βέλτιστη δρομολόγηση πλοίων ανάμεσα σε ένα σύνολο ενεργειών φόρτωσης και εκφόρτωσης ενός προκαθορισμένου συνόλου φορτίων, έτσι ώστε να ελαχιστοποιηθεί το κόστος εξυπηρέτησης των συγκεκριμένων φορτίων. Τα συστήματα αυτά είναι λιγοστά στην αγορά των ναυτιλιακών ηλεκτρονικών προγραμμάτων καθώς έχουν αναπτυχθεί από λιγοστές ναυτιλιακές εταιρείες οι οποίες δραστηριοποιούνται στο χώρο της βιομηχανικής αγοράς.

Ηλεκτρονικές αγορές: Τα τελευταία χρόνια αυξάνεται η χρήση των ηλεκτρονικών αγορών και στην διαδικασία ναύλωσης. Για την ακρίβεια οι ενδιαφερόμενοι για μεταφορά φορτίου μέσω θαλάσσης, καταχωρούν τα στοιχεία για κάθε φορτίο προς μεταφορά σε συγκεκριμένους δικτυακούς τόπους, από όπου οι εγγεγραμμένες ναυτιλιακές εταιρείες μπορούν να αναζητήσουν φορτία προς μεταφορά. Συχνά τέτοια συστήματα μετατρέπονται σε δικτυακούς πλειστηριασμούς όπου κάθε εταιρεία καταθέτει την δική της προσφορά για το φορτίο που ενδιαφέρεται και ο κάτοχος του φορτίου αποφασίζει για την προσφορά την οποία θα δεχθεί. Υπάρχουν

περιπτώσεις όπου δίνεται στις ναυτιλιακές εταιρείες η δυνατότητα να καταγράψουν στο σύστημα τα πλοία που διαθέτουν και να ενημερώνουν για την θέση των πλοίων και την κατάσταση τους. Με αυτό τον τρόπο οι ηλεκτρονικές αγορές κάνουν αντιστοίχηση πλοίων και φορτίων και παρουσιάζουν στις εταιρείες ένα σύνολο από φορτία που είναι εφικτό να μεταφερθούν από ένα συγκεκριμένο πλοίο που θα βρίσκεται σε μια δεδομένη περιοχή για μια ορισμένη χρονική περίοδο. Σε ορισμένες από αυτές τις ηλεκτρονικές αγορές παρέχονται εργαλεία ανάλυσης κόστους ενός ταξιδιού έτσι ώστε να διευκολύνουν τις ναυτιλιακές εταιρείες στην επιλογή της πιο συμφέρουσας μεταφοράς φορτίου.

Όπως γίνεται αντιληπτό η διαδικασία της ναύλωσης υποστηρίζεται με ηλεκτρονικά μέσα, κυρίως στο στάδιο της ενημέρωσης των διαθέσιμων φορτίων και σε περιορισμένες περιπτώσεις στο στάδιο του διαχωρισμού των φορτίων που μπορούν να μεταφερθούν από ένα συγκεκριμένο πλοίο. Η εύρεση των πιο κερδοφόρων διαθέσιμων φορτίων στηρίζεται, λόγο του όγκου των δεδομένων και των ελέγχων που πρέπει να πραγματοποιηθούν, στην διεξοδική μελέτη ενός μόνο συνόλου διαθέσιμων φορτίων. Η επιλογή του συνόλου αυτού των φορτίων που αξιολογούνται γίνεται εμπειρικά χωρίς την χρήση πληροφοριακών συστημάτων ενώ και η ίδια η αξιολόγηση των φορτίων που προκύπτουν στηρίζεται σε μεγάλο βαθμό στην εμπειρία των ατόμων που την πραγματοποιούν με μοναδική ηλεκτρονική υποστήριξη τα πακέτα ανάλυσης κόστους.

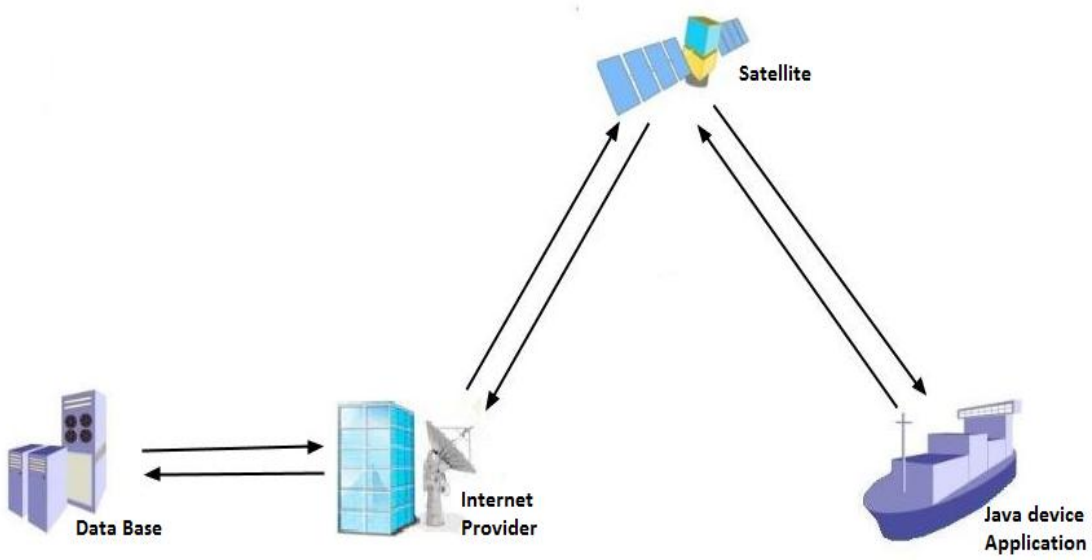
Κεφάλαιο 3

Σχεδιασμός & Υλοποίηση

Η παρούσα πτυχιακή εργασία αναπτύχθηκε με σκοπό την διαχείριση πληρώματος πλοίων. Πιο Συγκεκριμένα η εφαρμογή που υλοποιήθηκε στοχεύει στην διαχείριση των προσωπικών στοιχείων των μελών του πληρώματος καθώς και την αυτοματοποίηση της διαδικασίας συμπλήρωσης εγγράφων τύπου xls.

Ειδικότερα, δίνεται η δυνατότητα στον χρήστης να ελέγξει και να επεξεργαστεί τα στοιχεία του πλοίου και του πληρώματος. Πιο αναλυτικά ο χρήστης βλέπει τα δεδομένα του πληρώματος σε πίνακες και μπορεί πολύ εύκολα να τα επεξεργαστεί, απλά επιλέγοντας κάποιο μέλος από τους πίνακες. Πέρα του ελέγχου και τις επεξεργασίας, η εφαρμογή που υλοποιήσαμε προσφέρει τις λειτουργίες προσθήκη και διαγραφή μελών του προσωπικού του πλοίου. Επίσης μια ακόμα δυνατότητα που προσφέρεται στον χρήστη είναι η αυτοματοποιημένη δημιουργία εγγράφων τύπου xls επιλέγοντας τα. Τα προαναφερθέντα έγγραφα αφορούν κατά κανόνα τα στοιχεία του πλοίου καθώς και τα προσωπικά στοιχεία των μελών του πληρώματος του πλοίου. Τέλος αξίζει να σημειωθεί ότι τα έγγραφα αυτά αποτελούν βασική προϋπόθεση για την άφιξη των πλοίων στα λιμάνια προορισμού τους.

Η υλοποίηση της εφαρμογής διακρίνεται σε 2 τμήματα: την βάση δεδομένων και την java εφαρμογή με γραφικό περιβάλλον για τον χρήστη εικόνα 3.01.



Εικόνα 3.01 Αρχιτεκτονική Συστήματος

3.1 Βάση Δεδομένων

Στην βάση δεδομένων που δημιουργήσαμε αποθηκεύονται όλα τα δεδομένα του πλοίου και του πληρώματος. Στα πλαίσια της παρούσας πτυχιακής εργασίας χρησιμοποιήθηκε η βάση ανοιχτού κώδικα MySQL, και εργαλείο phpMyAdmin για τον σχεδιασμό, τη δημιουργία και την παραμετροποίηση της βάσης δεδομένων.

Ακολουθούν οι πίνακες της βάσης δεδομένων:

Πίνακας Vessel: Περιέχει όλα τα απαραίτητα γνωρίσματα για το πλοίο.

Fields Name	Type	Length	Description
ShipsName	Varchar	50	SHIP'S NAME
CallSign	Varchar	50	CALL SIGN
ImoNumber	Varchar	50	IMO NUMBER
OfficialNumber	Varchar	50	OFFICIAL NUMBER
Nationality	Varchar	50	NATIONALITY
TypeOfVessel	Varchar	50	TYPE OF VESSEL
PortOfRegistry	Varchar	50	PORT OF REGISTRY
CountryOfRegistry	Varchar	50	COUNTRY OF REGISTRY
GrossTonnage	Varchar	50	GROSS TONNAGE
NetTonnage	Varchar	50	NET TONNAGE
DeadWeight	Varchar	50	DEAD WEIGHT
Owners	Varchar	50	OWNERS
Operators	Varchar	50	OPERATORS
VoyageNumber	Varchar	50	VOYAGE NUMBER
MastersName	Varchar	50	MASTER'S NAME
SsoName	Varchar	50	SSO NAME
Charterers	Varchar	50	CHARTERERS
NumberOfCrew	Varchar	50	NUMBER OF CREW
CargoMT	Varchar	50	CARGO (MT)
CompanyIdentificationNo	Varchar	50	COMPANY IDENTIFICATION No
OwnersRegisteredNo	Varchar	50	OWNERS REGISTERED No

Πίνακας LogIn: Περιέχει απαραίτητα στοιχεία για να συνδεθεί ο χρήστης στην εφαρμογή.

Fields Name	Type	Length	Description
UserName	Varchar	50	USER NAME
Password	Varchar	50	PASSWORD

Πίνακας CrewMembers: Περιέχει τα περισσότερα απο τα προσωπικά δεδομένα των μελών του πληρώματος. Όταν προστίθενται νέα μέλη στο πλήρωμα δημιουργούνται καινούργιες εγγραφές στον συγκεκριμένο πίνακα. Αντίστοιχα, όταν αφαιρούνται μέλη, οι εγγραφές αφαιρούνται και όταν επεξεργαζόμαστε ανανεώνονται. Κάθε εγγραφή προσδιορίζεται μοναδικά από το πεδίο usaVisaFoil.

Fields Name	Type	Length	Description
familyName	Varchar	50	FAMILY NAME
firstName	Varchar	50	FIRST NAME
middleName	Varchar	50	MIDDLE NAME
middleInitial	Varchar	50	MIDDLE INITIAL
rank	Varchar	50	RANK
dateOfBirth	Varchar	50	DATE OF BIRTH
placeOfBirth	Varchar	50	PLACE OF BIRTH
countryOfBirth	Varchar	50	COUNTRY OF BIRTH
nationality	Varchar	50	NATIONALITY
signedOnPlace	Varchar	50	SIGNED ON PLACE
countryOfSignOn	Varchar	50	COUNTRY OF SIGN ON

signOnDate	Varchar	50	SIGNED ON DATE
lisenceNo	Varchar	50	LISENCE NO
expiryDate	Varchar	50	EXPIRY DATE
sex	Varchar	50	SEX
age	Varchar	50	AGE
yellowFeverIssue	Varchar	50	YELLOW FEVER ISSUE
yellowFeverExpire	Varchar	50	YELLOW FEVER EXPIRE
usd	Varchar	50	USD
euro	Varchar	50	EURO
various	Varchar	50	VARIOUS
spirits	Varchar	50	SPIRITS
cigarettes	Varchar	50	CIGARETTES
tobacco	Varchar	50	TOBACCO
personalThings	Varchar	400	PERSONAL THING
choleraIssue	Varchar	50	CHOLERA ISSUE
choleraExpire	Varchar	50	CHOLERA EXPIRE
beerCases	Varchar	50	BEER (CASES)
usaVisaExpDate	Varchar	50	USA VISA EXP. DATE
fatherName	Varchar	50	FATHER'S NAME
motherName	Varchar	50	MOTHER'S NAME
issueCountryOfSBfotSanCrewL	Varchar	50	ISSUE COUNTRY OF S.B
usaVisaFoil	Varchar	50	USA VISA FOIL – PRIMARY KEY
usaVisaControl	Varchar	50	USA VISA CONTROL
countryOfResidence	Varchar	50	COUNTRY OF RESIDENCE

cityOfResidence	Varchar	50	CITY OF RESIDENCE
seniorityInCompany	Varchar	50	SENIORITY IN COMPANY

Πίνακας SeamansBook: Περιέχει τα απαραίτητα στοιχεία για τα ναυτικά φυλλάδια των μελών του πληρώματος. Κάθε εγγραφή προσδιορίζεται μοναδικά από το πεδίο usaVisaFoil.

Fields Name	Type	Length	Description
No	Varchar	50	SEAMAN BOOK NUMBER
IssuedPlace	Varchar	50	ISSUED PLACE
ExpiryDate	Varchar	50	EXPIRY DATE
UsaVisaFoil	Varchar	50	USA VISA FOIL – PRIMARY KEY

Πίνακας Passport: Περιέχει τα απαραίτητα στοιχεία για τα διαβατήρια των μελών του πληρώματος. Κάθε εγγραφή προσδιορίζεται μοναδικά από το πεδίο usaVisaFoil.

Fields Name	Type	Length	Description
No	Varchar	50	PASSPORT NUMBER
IssuedPlace	Varchar	50	ISSUED PLACE
ExpiryDate	Varchar	50	EXPIRY DATE
UsaVisaFoil	Varchar	50	USA VISA FOIL – PRIMARY KEY

3.2 Εφαρμογή Java

Η ανάπτυξη της εφαρμογής πραγματοποιήθηκε στο NeatBeans. Το NeatBeans είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης ανοιχτού λογισμικού (Integrated Development Environment - IDE), χρησιμοποιώντας την γλώσσα προγραμματισμού Java. Η εφαρμογή αναπτύχθηκε κάνοντας κυρίως χρήση των προ εγκατεστημένων βιβλιοθηκών του NeatBeans και κάποιων επιπρόσθετων βιβλιοθηκών για την καλύτερη εμφάνιση της εφαρμογής αλλά και για την δημιουργία των εγγράφων.

Σχετικά με την υλοποίηση του κώδικα της εφαρμογής , έχουν υλοποιηθεί τέσσερις κλάσεις για την δημιουργία των οντοτήτων (μέλος πληρώματος, πλοίο, ναυτικό φυλλάδιο και διαβατήριο), μια κλάση η οποία αναλαμβάνει όλες τις συνδέσεις με την βάση δεδομένων, μια κλάση η οποία είναι υπεύθυνη για την δημιουργία αντικειμένων και λιστών που στην συνέχεια προβάλλονται στον χρήστη μέσω πινάκων. Τρεις ακόμα κλάσεις υπάρχουν για την δημιουργία των πινάκων που προβάλουν το σύνολο των δεδομένων του προσωπικού (ένας πίνακας με τα περισσότερα στοιχεία του πληρώματος, ένας πίνακας με τα στοιχεία για τα ναυτικά φυλλάδια και ένας πίνακας για τα διαβατήρια). Επίσης έχουμε δημιουργήσει μια κλάση η οποία δημιουργεί και εμφανίζει τις περισσότερες οθόνες της εφαρμογής στον χρήστη. Συγκεκριμένα η κλάση αυτή αναλαμβάνει να εμφανίσει όλες τις οθόνες της εφαρμογής εκτός της οθόνης για την δημιουργία των εγγράφων. Την εμφάνιση της οθόνης για τα xls έγγραφα, όπως και την δημιουργία τους, την πραγματοποιεί μια ακόμη κλάση. Από τις κλάσεις που προαναφέρθηκαν, η σημαντικότερη είναι η MainView που είναι υπεύθυνη για την προβολή του μεγαλύτερου μέρους των οθονών στον χρήστη. Η MainView καλεί όλες τις υπόλοιπες κλάσεις και δημιουργεί τα επιθυμητά αποτελέσματα. Ειδικότερα όπως βλέπουμε στην εικόνα 3.02 στο τμήμα κώδικα που εκτελείται, όταν ο χρήστης προσπαθεί να συνδεθεί στην εφαρμογή, καλείται η κλάση db_connection και η κατάλληλη συνάρτηση της για των έλεγχο των διαπιστευτηρίων. Στην συνέχεια αν η σύνδεση ήταν επιτυχής καλούνται οι κλάσεις TableModelClass, SeamansBookJtableModel, PassportJtableModel προκειμένου να δημιουργηθούν οι

πίνακες προβολής των δεδομένων του πληρώματος αφού ο χρήστης θα μεταβεί στην κεντρική οθόνη της εφαρμογής.

```
boolean formCompleted = true;
wrongLoginLabel.setText("");

if (userNameTxtArea.getText().isEmpty() || passwordTxtArea.getText().isEmpty()) {
    formCompleted = false;
    wrongLoginLabel.setText("Username and Password required!");
}

if (formCompleted) {
    db_connection connectDB = new db_connection();
    connectDB.connect();

    String userName = userNameTxtArea.getText();
    String password = passwordTxtArea.getText();

    int loginSuc_userID = connectDB.checkLogin(userName, password);

    if (loginSuc_userID > 0) {
        LogInPanel.setVisible(false);
        CrewMemTabbedPane2.setVisible(true);
        buttonPanel1.setVisible(true);
        EditCrewPanel.setVisible(false);
        editVesselInfoButton.setVisible(false);

        jTable1.setModel(new TableModelClass());
        SeamansBookJtable.setModel(new SeamansBookJtableModel());
        passportsJtable.setModel(new PassportJtableModel());

        makeColumnsCrewTable();
        makeColumnsSeamansBookTable();
        makeColumnsPassportTable();
    } else {
        wrongLoginLabel.setText("Wrong Username or Password!");
    }
    connectDB.closeConnection();
}
```

Εικόνα 3.02 Τμήμα κώδικα κατά την προσπάθεια σύνδεσης του χρήστη

Όπως έχουμε αναφέρει η εφαρμογή που αναπτύξαμε διαθέτει στον χρήστη τις δυνατότητες προβολής και επεξεργασίας για τα στοιχεία του πλοίου και προβολής, επεξεργασίας, προσθήκης και διαγραφής των δεδομένων του πληρώματος. Όταν ο χρήστης συνδεθεί στην εφαρμογή έχει την δυνατότητα να εκτελέσει οποιαδήποτε από τις παραπάνω λειτουργίες. Όλη η αλληλεπίδραση γίνεται μέσω των οθονών που δημιουργούνται από την κλάση MainView. Συγκεκριμένα, αν ο χρήστης επιλέξει να επεξεργαστεί κάποιο μέλος του προσωπικού τότε όπως βλέπουμε στο τμήμα του κώδικα της εικόνας 3.03, καλείτε η κλάση db_connection ώστε με την χρήση

των κατάλληλων συναρτήσεων που έχουμε δημιουργήσει σε αυτή, λαμβάνουμε το σύνολο των προσωπικών δεδομένων του και απ τους τρεις πίνακες της βάσης δεδομένων, που αφορούν τα στοιχεία του πληρώματος. Στην συνέχεια τα δεδομένα του μέλους που επιλέχθηκε από τον χρήστη για επεξεργασία τοποθετούνται σε κατάλληλα πεδία με σκοπό την διευκόλυνση και την πρόληψη λαθών του χρήστη, λόγο του μεγάλου όγκου δεδομένων ανά μέλος πληρώματος.

```
db_connection conDB = new db_connection();
conDB.connect();

crewMem = conDB.getCrewMembers();
seamans = conDB.getSeamansBook();
pass = conDB.getPassport();

SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
Date date = new Date();


int row = jTable1.getSelectedRow();
Dimension size = EdtabaccoTextField20.size();
EdmiddleInitialTextField5.setPreferredSize(size);
EdfamilyNameTextField2.setText(crewMem.get(row).getFamilyName());
EdfistNameTextField3.setText(crewMem.get(row).getFistName());
EdmiddleNameTextField4.setText(crewMem.get(row).getMiddleName());
EdmiddleInitialTextField5.setText(crewMem.get(row).getMiddleInitial());
EdrankTextField6.setText(crewMem.get(row).getRank());

try {

    date = format.parse(crewMem.get(row).getDateOFBirth());

    DateOfBirthljDateChooser.setDate(date);

} catch (ParseException ex) {
    Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE, null, ex);
}
```

Εικόνας 3.03 Τμήμα κώδικα κατά την επιλογή μέλους για επεξεργασία

Στην συνέχεια αφού η διαδικασία της επεξεργασίας από τον χρήστη ολοκληρωθεί επιτυχώς χωρίς την παράληψη συμπλήρωσης κάποιου υποχρεωτικού πεδίου, όπως βλέπουμε στο τμήμα κώδικα της εικόνας 3.04 πραγματοποιούνται κλήσεις των κλάσεων CrewMembers, SeamansBook, Vessel με σκοπό να δημιουργηθούν αντικείμενα με τα νέα δεδομένα που προέκυψαν απ την διαδικασία της επεξεργασίας. Ακόμα καλείται η db_connection και οι κατάλληλες συναρτήσεις της έτσι ώστε, να ανανεωθούν οι βάσεις δεδομένων μας με τα νέα στοιχεία των αντικείμενων που δημιουργήθηκαν.

```
String cityOfResidence = EdCityOfTextField25.getText();
String seniorityInCompany = EdSeInComTextField27.getText();

//Seaman's book
String NoSea = EdnoTextField5.getText();
String issuedPlaceSea = edisplTextField6.getText();
date = ExpirySeajDateChooser.getDate();
String expiryDateSea = String.format("%1$td/%1$tm/%1$tY", date);

//Passport
String No = Edno2TextField3.getText();
String issuedPlacePass = Edispl2TextField6.getText();
date = expiryPassjDateChooser.getDate();
String expiryDatePass = String.format("%1$td/%1$tm/%1$tY", date);

CrewMember member = new CrewMember(familyName, fistName, middleName, middleInitial, rank,
        countryOfSignOn, signOnDate, lisenceNo, expiryDate, sex, age, yellowFeverIssue, 
        choleraIssue, choleraExpire, beerCases, usaVisaExpDate, fatherName, motherName, 
        cityOfResidence, seniorityInCompany);


SeamansBook  seam = new SeamansBook(NoSea,issuedPlacePass,expiryDateSea,usaVisaFoil);
Passport passport= new Passport(No,issuedPlacePass,expiryDatePass,usaVisaFoil );


db_connection dataBaseConnection = new db_connection();
dataBaseConnection.connect();

boolean successfulUpdateCrewMember = dataBaseConnection.updateCrewmemberData(member);

boolean  successUpdateSeamansBook= dataBaseConnection.updateSeamansBookData(seam);

boolean successUpdatePasspoprt= dataBaseConnection.updatePassportData(passport);
```

Εικόνας 3.04 Τμήμα κώδικα κατά την επιτυχημένη επεξεργασία μέλους του πληρώματος

Όταν ο χρήστης επιλέξει απ το μενού της εφαρμογής την δημιουργία εγγράφων τότε θα μεταβεί στην κλάση PortDocuments. Η PortDocuments καλεί την db_connection με τις αντίστοιχες συναρτήσεις της για την δημιουργία λιστών (ArrayLists) με όλα τα διαθέσιμα στοιχεία του πλοίου και του πληρώματος που έχουμε στην βάση δεδομένων μας. Οι λίστες που δημιουργήσαμε χρησιμεύουν στην συμπλήρωση των εγγράφων τύπου xls. Αξίζει να σημειωθεί ότι για την συμπλήρωση των εγγράφων κάναμε χρήση της βιβλιοθήκης Apache POI.

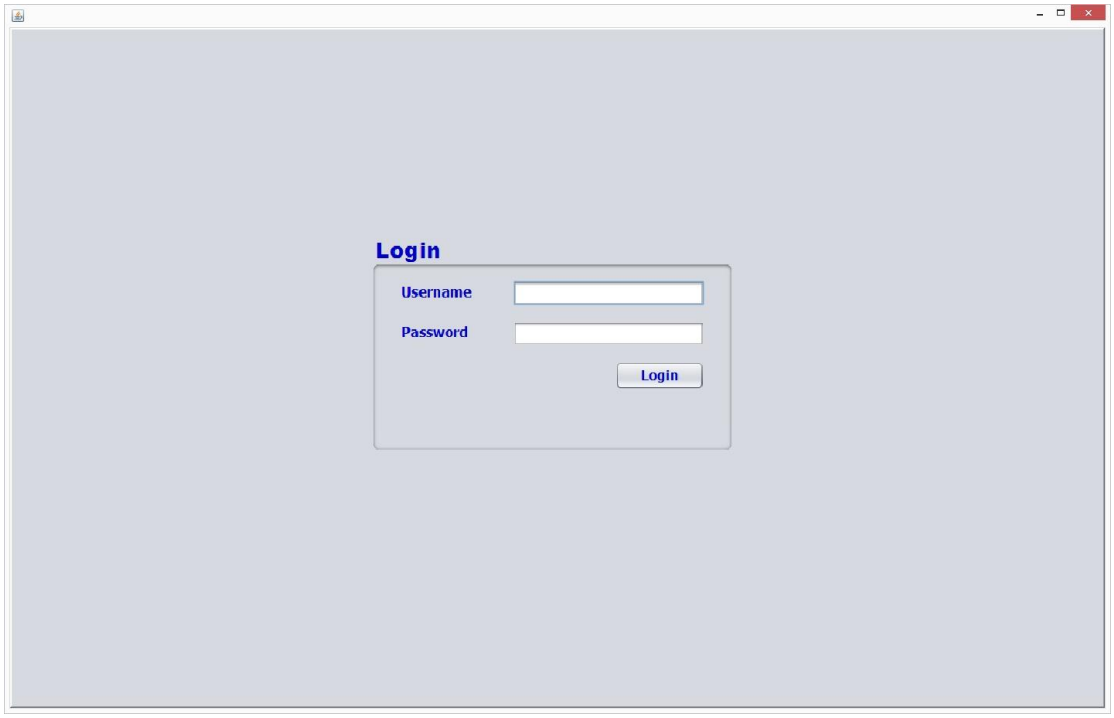
```
CrewDataDBLists CrewDataDBDb = new CrewDataDBLists();
db_connection condb = new db_connection();
condb.connect();

crewMem = CrewDataDBDb.GetCrewMembers();
passportList = CrewDataDBDb.GetPassport();
vesselList = condb.getVesselDB();

condb.closeConnection();
boolean success = (new File("C:/JavaApplication5/port")).mkdirs();

try {

    workbook = new HSSFWorkbook(new FileInputStream("C:\\\\JavaApplication5\\\\Excels\\\\CrewList2.xls"));

    Sheet sheet = workbook.getSheet("crew");

    for (int i = 0; i < crewMem.size(); i++) {
        Cell cellCount = sheet.getRow(i + 10).getCell(0);
        cellCount.setCellValue(i + 1);

        Row row = sheet.getRow(10 + i);
        Cell cell = row.getCell(1);
        cell.setCellValue(crewMem.get(i).getFamilyName().toString());

        Cell cellFirtName = sheet.getRow(i + 10).getCell(2);
        cellFirtName.setCellValue(crewMem.get(i).getFistName().toString());

        Cell cellMiddleName = sheet.getRow(i + 10).getCell(3);
        cellMiddleName.setCellValue(crewMem.get(i).getMiddleName().toString());

        Cell cellSex = sheet.getRow(i + 10).getCell(4);
        cellSex.setCellValue(crewMem.get(i).getSex());

        Cell cellRank = sheet.getRow(i + 10).getCell(5);
        cellRank.setCellValue(crewMem.get(i).getRank());
```

Εικόνα 3.05 Τμήμα κώδικα από την δημιουργία εγγράφων τύπου xls

Κεφάλαιο 4

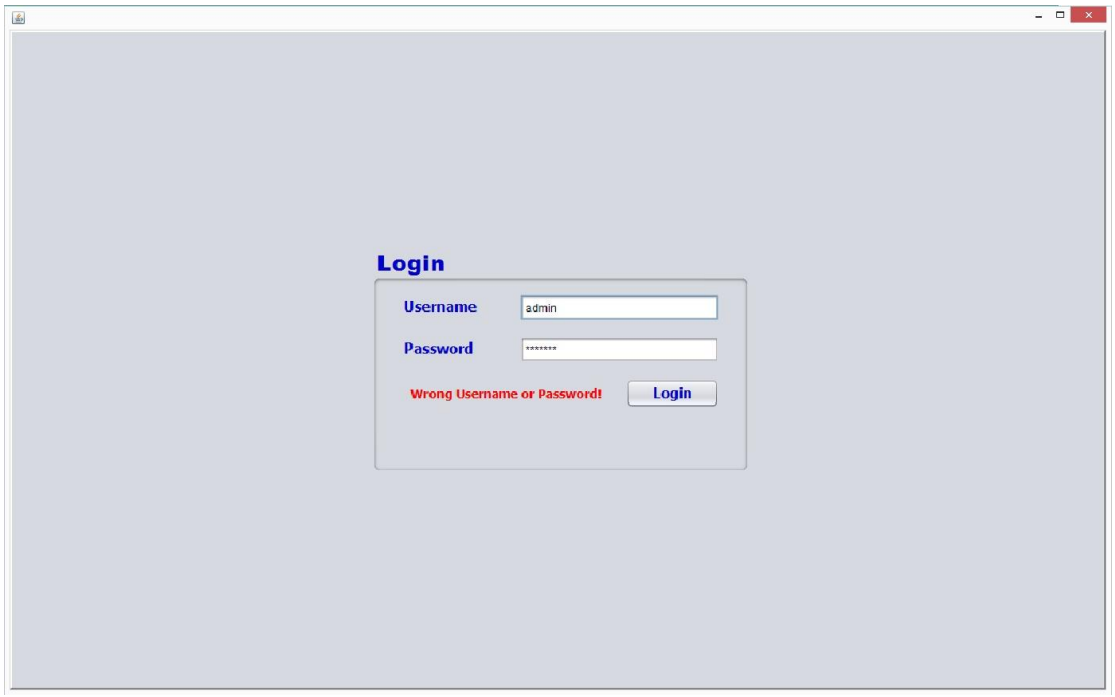
Παρουσίαση συστήματος και περιγραφή

Η αρχική οθόνη της εφαρμογής είναι για την είσοδο του χρήστη.



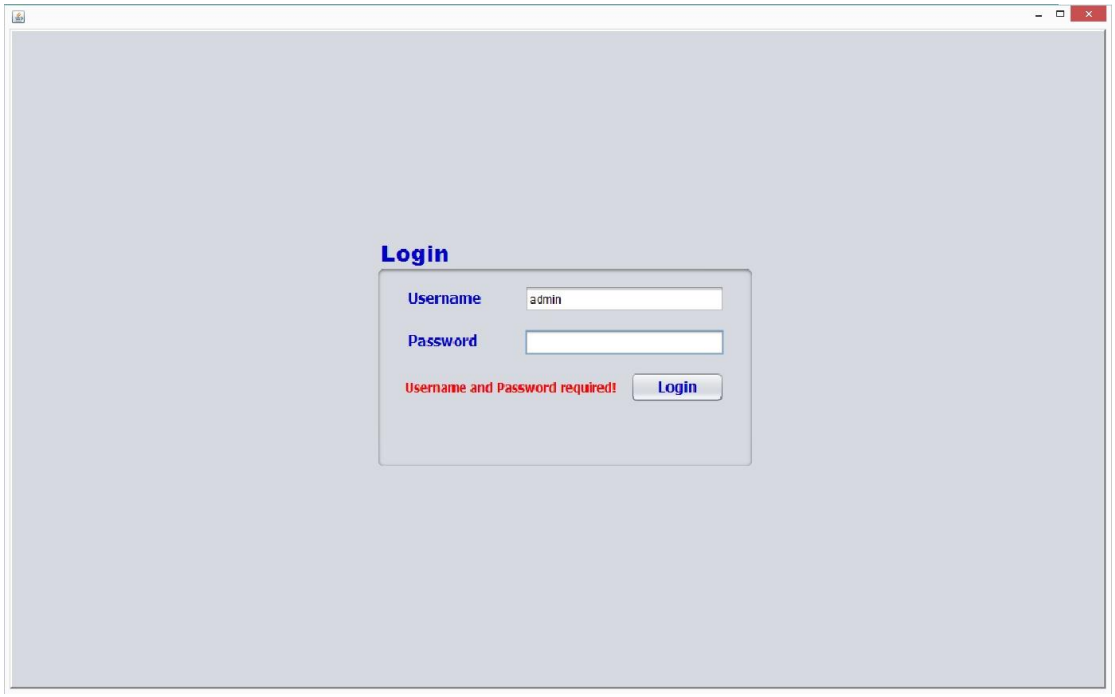
Εικόνα 4.01 Οθόνη σύνδεσης

Ο ηλεκτρονικός υπολογιστής όπου βρίσκεται η εφαρμογή, πρέπει να έχει πρόσβαση στο διαδίκτυο προκειμένου ο χρήστης της εφαρμογής να μπορεί να συνδεθεί. Αυτό επιτυγχάνεται μέσω δορυφορικού δικτύου που υπάρχει σε όλα τα σύγχρονα εμπορικά πλοία.

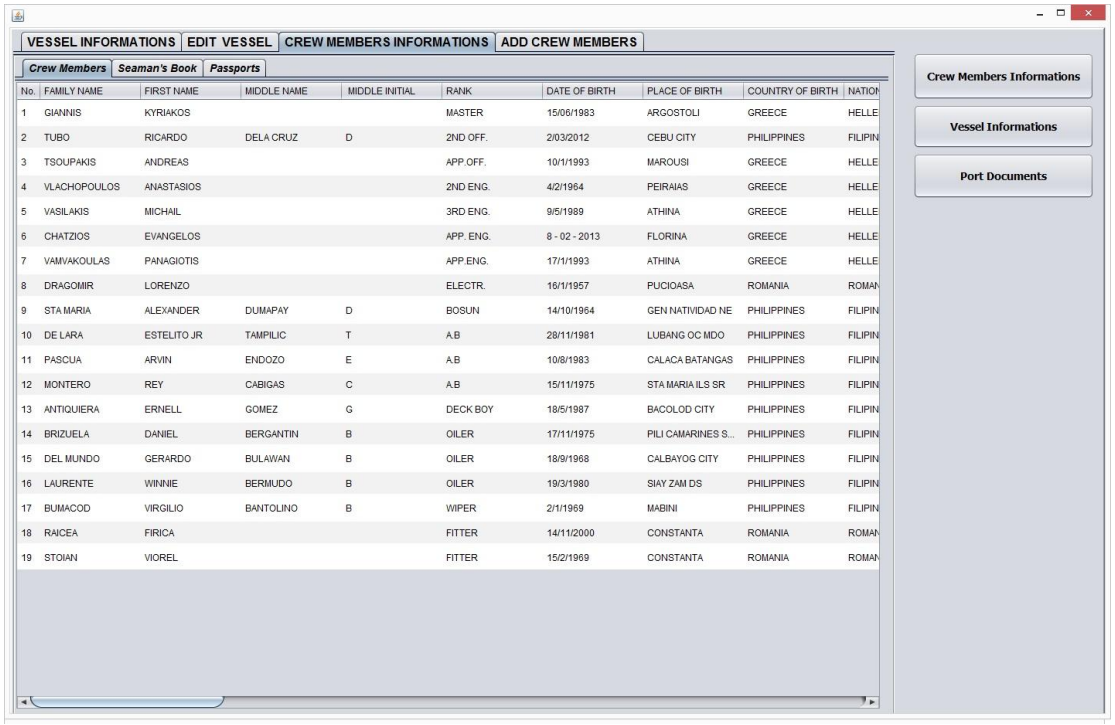


Εικόνα 4.02 Οθόνη σύνδεσης, μηνύματα για λάθος διαπιστευτήρια

Κατά την προσπάθεια εισόδου του χρήστη στην εφαρμογή, σε περίπτωση που τα διαπιστευτήρια του χρήστη είναι λανθασμένα, είτε το πεδίο username, είτε το πεδίο password εμφανίζεται μήνυμα λάθους. Επίσης αν κάποιο από τα πεδία διαπιστευτηρίων μείνει κενό εμφανίζεται αντίστοιχο μήνυμα λάθους εικόνα 4.03.



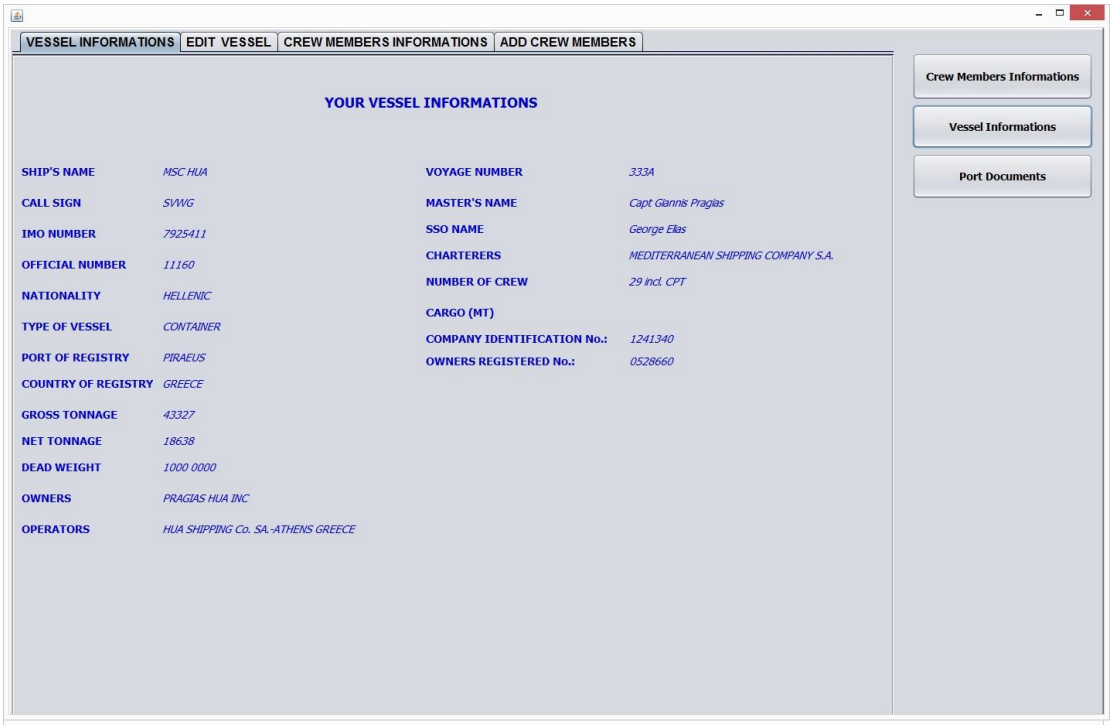
Εικόνα 4.03 Οθόνη σύνδεσης, μηνύματα για παράληψη συμπλήρωσης



No.	FAMILY NAME	FIRST NAME	MIDDLE NAME	MIDDLE INITIAL	RANK	DATE OF BIRTH	PLACE OF BIRTH	COUNTRY OF BIRTH	NATION
1	GIANNIS	KYRIAKOS			MASTER	15/06/1983	ARGOSTOLI	GREECE	HELLE
2	TUBO	RICARDO	DELA CRUZ	D	2ND OFF.	2/03/2012	CEBU CITY	PHILIPPINES	FILIPIN
3	TSOUPAKIS	ANDREAS			APP.OFF.	10/1/1993	MAROUSI	GREECE	HELLE
4	VLACHOPOULOS	ANASTASIOS			2ND ENG.	4/2/1964	PEIRAIAS	GREECE	HELLE
5	VASILAKIS	MICHAIL			3RD ENG.	9/5/1989	ATHINA	GREECE	HELLE
6	CHATZIOS	EVANGELOS			APP. ENG.	8 - 02 - 2013	FLORINA	GREECE	HELLE
7	VAMVAKOULAS	PANAGIOTIS			APP.ENG.	17/1/1993	ATHINA	GREECE	HELLE
8	DRAGOMIR	LORENZO			ELECTR.	16/1/1957	PUCIOASA	ROMANIA	ROMAN
9	STA MARIA	ALEXANDER	DUMAPAY	D	BOSUN	14/10/1964	GEN NATIVIDAD NE	PHILIPPINES	FILIPIN
10	DE LARA	ESTELITO JR	TAMPILIC	T	A.B	28/11/1981	LUBANG OC MDO	PHILIPPINES	FILIPIN
11	PASCUA	ARVIN	ENDOZO	E	A.B	10/8/1983	CALACA BATANGAS	PHILIPPINES	FILIPIN
12	MONTERO	REY	CABIGAS	C	A.B	15/11/1975	STA MARIA ILS SR	PHILIPPINES	FILIPIN
13	ANTIQUIERA	ERNELL	GOMEZ	G	DECK BOY	18/5/1987	BACOLOD CITY	PHILIPPINES	FILIPIN
14	BRIZUELA	DANIEL	BERGANTIN	B	OILER	17/11/1975	PILI CAMARINES S...	PHILIPPINES	FILIPIN
15	DEL MUNDO	GERARDO	BULAWAN	B	OILER	18/9/1968	CALBAYOG CITY	PHILIPPINES	FILIPIN
16	LAURENTE	WINNIE	BERMUDO	B	OILER	19/3/1980	SIAY ZAM DS	PHILIPPINES	FILIPIN
17	BUMACOD	VIRGILIO	BANTOLINO	B	WIPER	2/1/1969	MABINI	PHILIPPINES	FILIPIN
18	RAICEA	FIRICA			FITTER	14/11/2000	CONSTANTA	ROMANIA	ROMAN
19	STOIAN	VIOREL			FITTER	15/2/1969	CONSTANTA	ROMANIA	ROMAN

Εικόνα 4.04 Εκκίνηση εφαρμογής

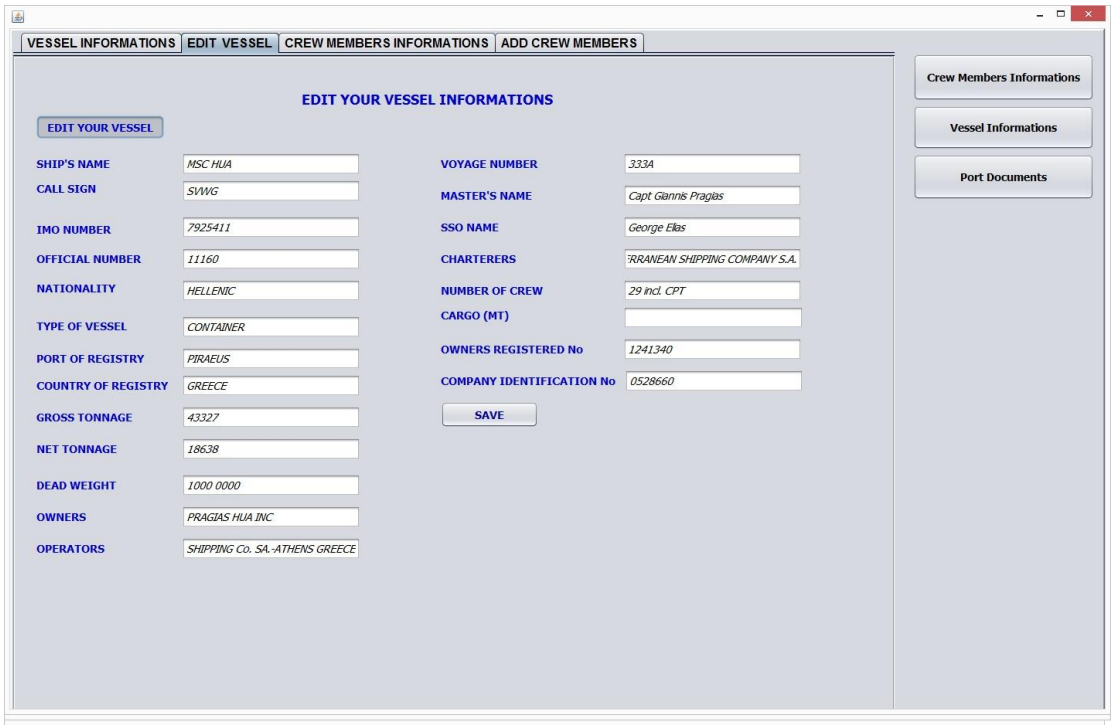
Στην περίπτωση που τα διαπιστευτήρια είναι ορθά, τότε μεταβαίνει στην κεντρική οθόνη. Σε αυτό το σημείο ο χρήστης έχει την δυνατότητα να ελέγξει τις πληροφορίες κάθε μέλους του πληρώματος καθώς και του πλοίου. Επίσης, μπορεί να επεξεργαστεί τα στοιχεία και τις πληροφορίες όλων των μελών του πληρώματος, όπως και του πλοίου. Ακόμα ο χρήστης έχει την δυνατότητα να προσθέσει και να διαγράψει καινούργια μέλη στο πλήρωμα. Τέλος η εφαρμογή μας προσφέρει την δυνατότητα αυτόματης συμπλήρωσης σε συγκεκριμένα έγγραφα τύπου xls. Τα συγκεκριμένα έγγραφα είναι απαραίτητα για την άφιξη του πλοίου στα λιμάνια προορισμού.



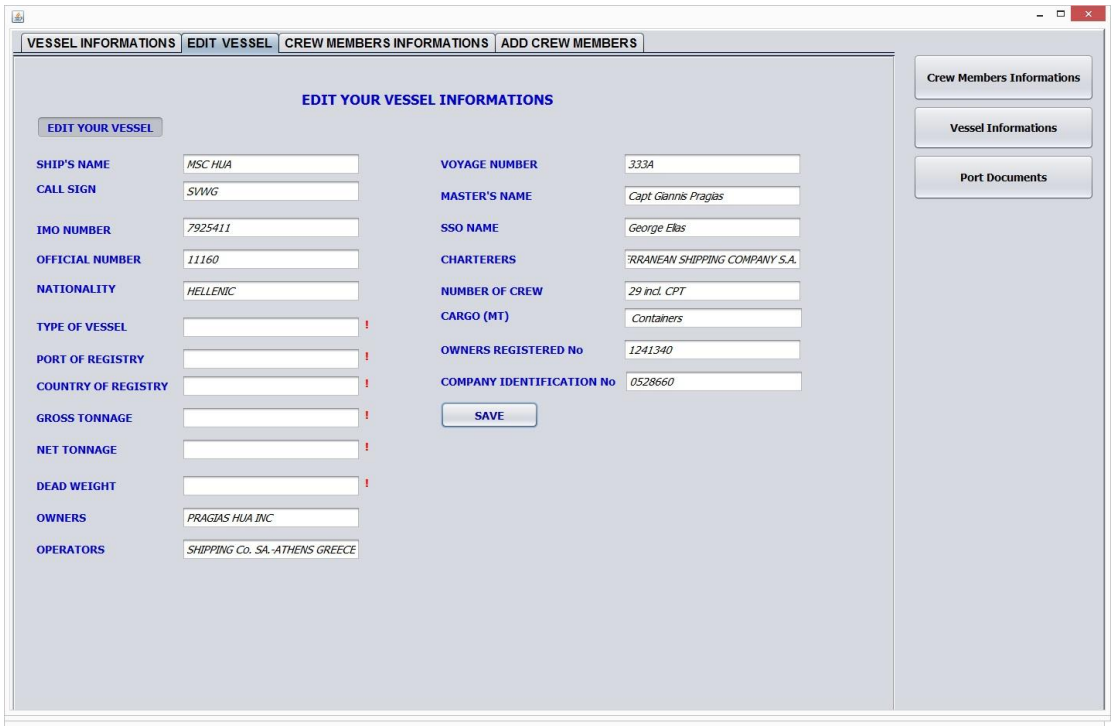
Εικόνα 4.05 Οθόνη προβολής πληροφοριών πλοίου

Αν επιλέξει την πρώτη καρτέλα της εφαρμογής μας δηλαδή να δει τις πληροφορίες του πλοίου τότε πιέζοντας το κουμπί Vessel Information’s οι πληροφορίες του πλοίου ανανεώνονται απ την βάση δεδομένων και εμφανίζονται. Στην συνέχεια αν ο χρήστης επιλέξει να επεξεργαστεί τις πληροφορίες του πλοίου τα πεδία των

στοιχείων του πλοίου συμπληρώνονται με τις υπάρχουσες πληροφορίες απ την βάση δεδομένων εικόνα 4.06.

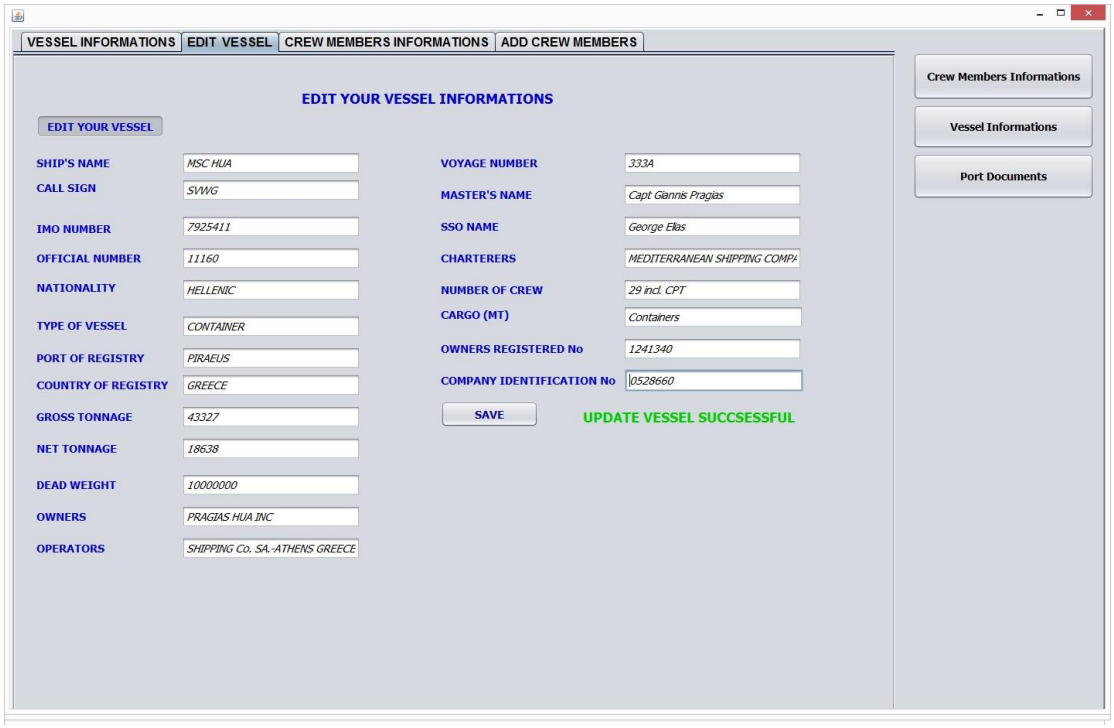


Εικόνα 4.06 Οθόνη επεξεργασίας πληροφοριών πλοίου

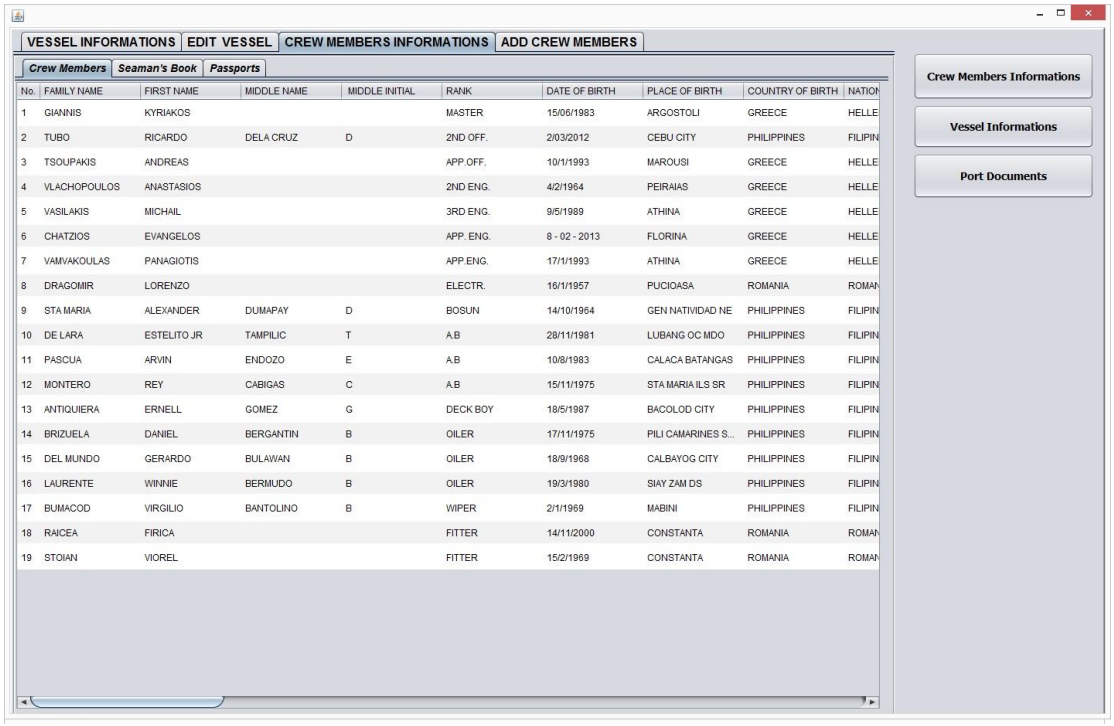


Εικόνα 4.07 Οθόνη επεξεργασίας πληροφοριών πλοίου μηνύματα λάθους

Όταν ολοκληρωθεί η επεξεργασία των πληροφοριών του πλοίου αν κάποιο από τα υποχρεωτικά πεδία έχει μείνει κενό ο χρήστης ειδοποιείται ώστε να συμπληρώσει τα αντίστοιχα πεδία. Στην περίπτωση που όλα τα υποχρεωτικά πεδία συμπληρωθούν τα καινούργια στοιχεία αποθηκεύονται στην βάση δεδομένων και ο χρήστης ενημερώνεται με το αντίστοιχο μήνυμα επιτυχίας εικόνα 4.08.



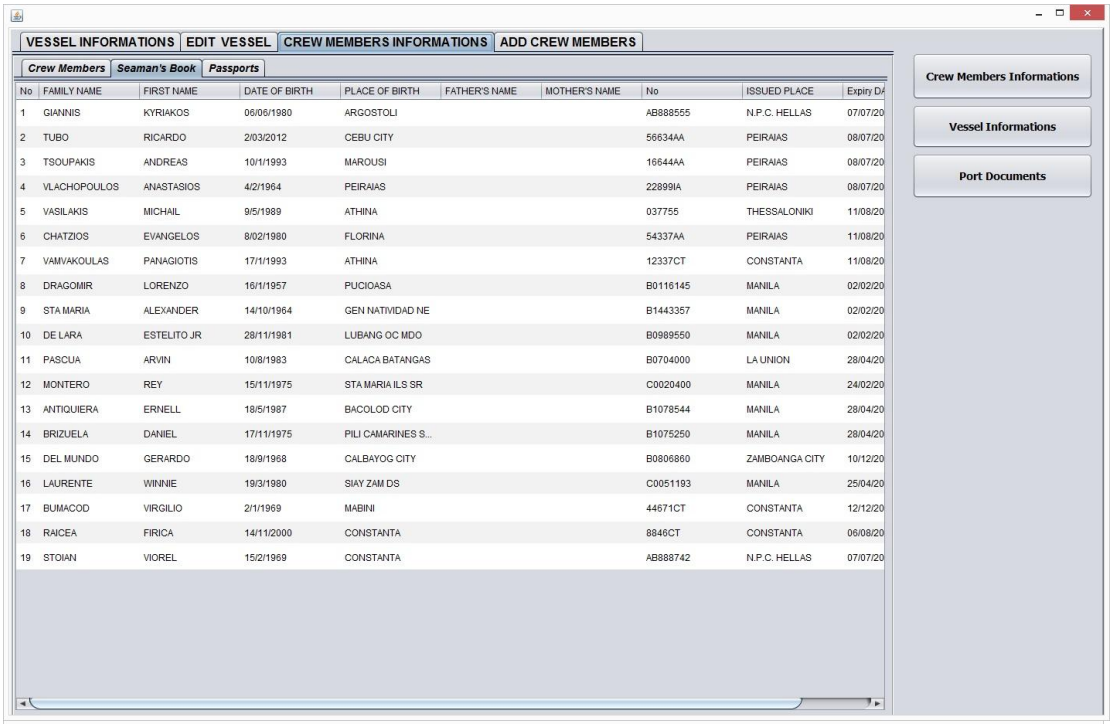
Εικόνα 4.08 Οθόνη επεξεργασίας πληροφοριών πλοίου επιτυχημένη καταχώρηση



No.	FAMILY NAME	FIRST NAME	MIDDLE NAME	MIDDLE INITIAL	RANK	DATE OF BIRTH	PLACE OF BIRTH	COUNTRY OF BIRTH	NATION
1	GIANNIS	KYRIAKOS			MASTER	15/06/1983	ARGOSTOLI	GREECE	HELLE
2	TUBO	RICARDO	DELA CRUZ	D	2ND OFF.	2/03/2012	CEBU CITY	PHILIPPINES	FILIPIN
3	TSOUPAKIS	ANDREAS			APP.OFF.	10/1/1993	MAROUSI	GREECE	HELLE
4	VLACHOPOULOS	ANASTASIOS			2ND ENG.	4/2/1964	PEIRAIAS	GREECE	HELLE
5	VASILAKIS	MICHAIL			3RD ENG.	9/5/1989	ATHINA	GREECE	HELLE
6	CHATZIOS	EVANGELOS			APP. ENG.	8 - 02 - 2013	FLORINA	GREECE	HELLE
7	VAMVAKOULAS	PANAGIOTIS			APP.ENG.	17/1/1993	ATHINA	GREECE	HELLE
8	DRAGOMIR	LORENZO			ELECTR.	16/1/1957	PUCIOASA	ROMANIA	ROMAN
9	STA MARIA	ALEXANDER	DUMAPAY	D	BOSUN	14/10/1964	GEN NATIVIDAD NE	PHILIPPINES	FILIPIN
10	DE LARA	ESTELITO JR	TAMPILIC	T	A.B	28/11/1981	LUBANG OC MDO	PHILIPPINES	FILIPIN
11	PASCUA	ARVIN	ENDOZO	E	A.B	10/8/1983	CALACA BATANGAS	PHILIPPINES	FILIPIN
12	MONTERO	REY	CABIGAS	C	A.B	15/11/1975	STA MARIA ILS SR	PHILIPPINES	FILIPIN
13	ANTIQUIERA	ERNELL	GOMEZ	G	DECK BOY	18/5/1987	BACOLOD CITY	PHILIPPINES	FILIPIN
14	BRIZUELA	DANIEL	BERGANTIN	B	OILER	17/11/1975	PILI CAMARINES S...	PHILIPPINES	FILIPIN
15	DEL MUNDO	GERARDO	BULAWAN	B	OILER	18/9/1968	CALBAYOG CITY	PHILIPPINES	FILIPIN
16	LAURENTE	WINNIE	BERMUDO	B	OILER	19/3/1980	SIAY ZAM DS	PHILIPPINES	FILIPIN
17	BUMACOD	VIRGILIO	BANTOLINO	B	WIPER	2/1/1969	MABINI	PHILIPPINES	FILIPIN
18	RAICEA	FIRICA			FITTER	14/11/2000	CONSTANTA	ROMANIA	ROMAN
19	STOIAN	VIOREL			FITTER	15/2/1969	CONSTANTA	ROMANIA	ROMAN

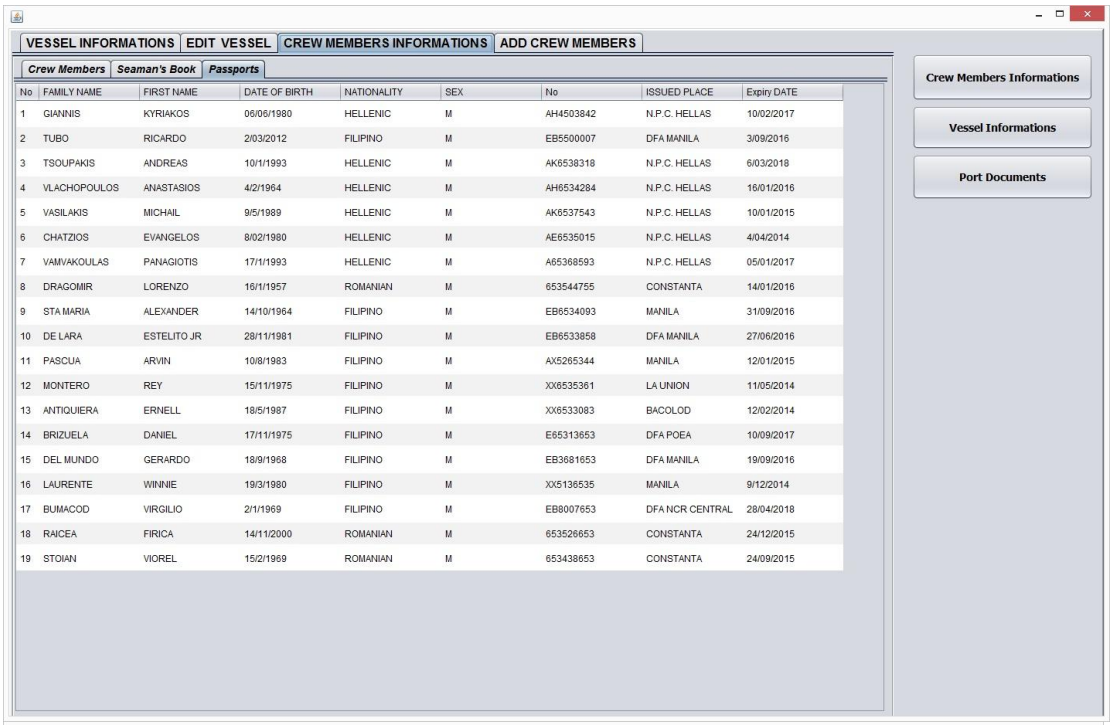
Εικόνα 4.09 Οθόνη προβολής γενικών πληροφοριών πληρώματος

Επιλέγοντας την καρτέλα εμφάνιση πληροφοριών πληρώματος, ο χρήστης έχει την δυνατότητα να δει το σύνολο των πληροφοριών του πληρώματος μέσω της υπό καρτέλας Crew Members. Επιπλέον μπορεί να ελέγξει ξεχωριστά τα στοιχεία των Ναυτικών Φυλλαδίων και των διαβατηρίων απ τις υπό καρτέλες Seaman’s Book και Passport αντίστοιχα εικόνες 4.10 και 4.11.



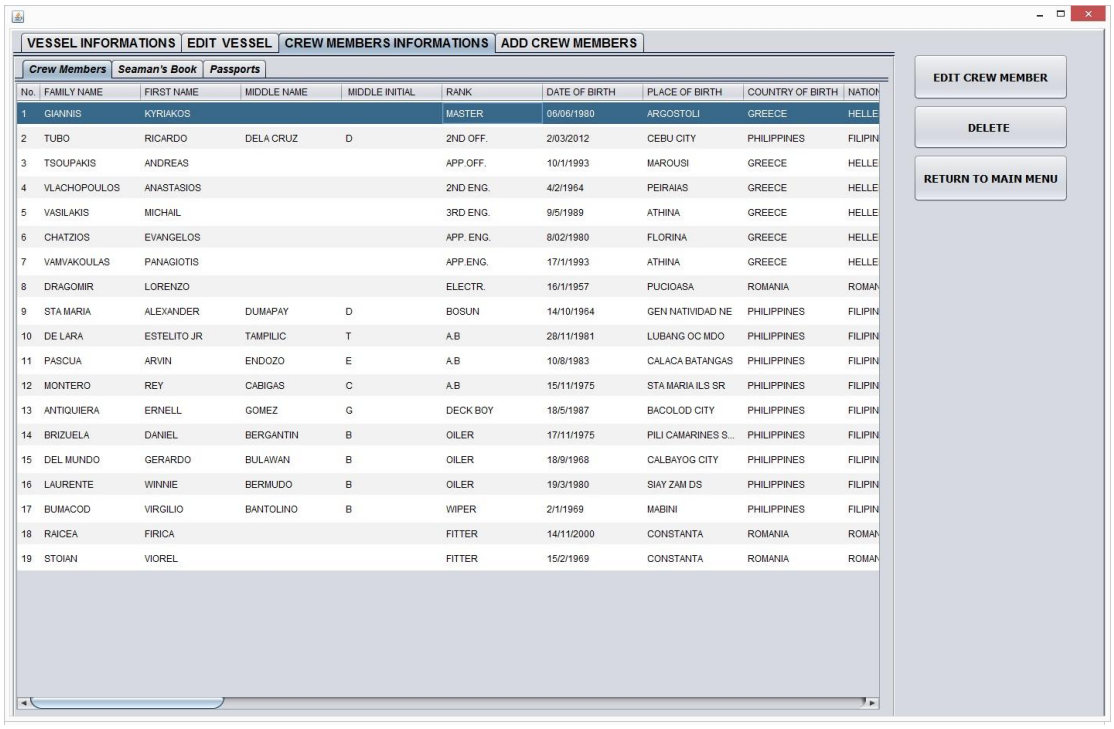
No	FAMILY NAME	FIRST NAME	DATE OF BIRTH	PLACE OF BIRTH	FATHER'S NAME	MOTHER'S NAME	No	ISSUED PLACE	Expiry DA
1	GIANNIS	KYRIAKOS	06/06/1980	ARGOSTOLI			AB888555	N.P.C. HELLAS	07/07/20
2	TUBO	RICARDO	2/03/2012	CEBU CITY			56634AA	PEIRAIAS	08/07/20
3	TSOUPAKIS	ANDREAS	10/1/1993	MAROUSI			16644AA	PEIRAIAS	08/07/20
4	VLACHOPOULOS	ANASTASIOS	4/2/1964	PEIRAIAS			22899IA	PEIRAIAS	08/07/20
5	VASILAKIS	MICHAIL	9/5/1989	ATHINA			037755	THESSALONIKI	11/08/20
6	CHATZIOS	EVANGELOS	8/02/1980	FLORINA			54337AA	PEIRAIAS	11/08/20
7	VAMVAKOULAS	PANAGIOTIS	17/1/1993	ATHINA			12337CT	CONSTANTA	11/08/20
8	DRAGOMIR	LORENZO	16/1/1957	PUCIOASA			B0116145	MANILA	02/02/20
9	STA MARIA	ALEXANDER	14/10/1964	GEN NATIVIDAD NE			B1443357	MANILA	02/02/20
10	DE LARA	ESTELITO JR	28/11/1981	LUBANG OC MDO			B0989550	MANILA	02/02/20
11	PASCUA	ARVIN	10/8/1983	CALACA BATANGAS			B0704000	LA UNION	28/04/20
12	MONTERO	REY	15/11/1975	STA MARIA ILS SR			C0020400	MANILA	24/02/20
13	ANTIQUIERA	ERNELL	18/5/1987	BACOLOD CITY			B1078544	MANILA	28/04/20
14	BRIZUELA	DANIEL	17/11/1975	PILI CAMARINES S...			B1075250	MANILA	28/04/20
15	DEL MUNDO	GERARDO	18/9/1968	CALBAYOG CITY			B0806860	ZAMBOANGA CITY	10/12/20
16	LAURENTE	WINNIE	19/3/1980	SIAY ZAM DS			C0051193	MANILA	25/04/20
17	BUMACOD	VIRGILIO	2/1/1969	MABINI			44671CT	CONSTANTA	12/12/20
18	RAICEA	FIRICA	14/11/2000	CONSTANTA			8846CT	CONSTANTA	06/08/20
19	STOIAN	VIOREL	15/2/1969	CONSTANTA			AB888742	N.P.C. HELLAS	07/07/20

Εικόνα 4.10 Οθόνη προβολής πληροφοριών ναυτικών φυλλαδίων



No	FAMILY NAME	FIRST NAME	DATE OF BIRTH	NATIONALITY	SEX	No	ISSUED PLACE	Expiry DATE
1	GIANNIS	KYRIAKOS	06/06/1980	HELLENIC	M	AH4503842	N.P.C. HELLAS	10/02/2017
2	TUBO	RICARDO	2/03/2012	FILIPINO	M	EB5500007	DFA MANILA	3/09/2016
3	TSOUPAKIS	ANDREAS	10/1/1993	HELLENIC	M	AK6538318	N.P.C. HELLAS	6/03/2018
4	VLACHOPOULOS	ANASTASIOS	4/2/1964	HELLENIC	M	AH6534284	N.P.C. HELLAS	16/01/2016
5	VASILAKIS	MICHAIL	9/5/1989	HELLENIC	M	AK6537543	N.P.C. HELLAS	10/01/2015
6	CHATZIOS	EVANGELOS	8/02/1980	HELLENIC	M	AE6535015	N.P.C. HELLAS	4/04/2014
7	VAMVAKOULAS	PANAGIOTIS	17/1/1993	HELLENIC	M	A65368593	N.P.C. HELLAS	05/01/2017
8	DRAGOMIR	LORENZO	16/1/1957	ROMANIAN	M	653544755	CONSTANTA	14/01/2016
9	STA MARIA	ALEXANDER	14/10/1964	FILIPINO	M	EB6534093	MANILA	31/09/2016
10	DE LARA	ESTELITO JR	28/11/1981	FILIPINO	M	EB6533858	DFA MANILA	27/06/2016
11	PASCUA	ARVIN	10/8/1983	FILIPINO	M	AX5265344	MANILA	12/01/2015
12	MONTERO	REY	15/11/1975	FILIPINO	M	XX6535361	LA UNION	11/05/2014
13	ANTIQUIERA	ERNELL	18/5/1987	FILIPINO	M	XX6533083	BACOLOD	12/02/2014
14	BRIZUELA	DANIEL	17/11/1975	FILIPINO	M	E65313653	DFA POEA	10/09/2017
15	DEL MUNDO	GERARDO	18/9/1968	FILIPINO	M	EB3681653	DFA MANILA	19/09/2016
16	LAURENTE	WINNIE	19/3/1980	FILIPINO	M	XX5136535	MANILA	9/12/2014
17	BUMACOD	VIRGILIO	2/1/1969	FILIPINO	M	EB8007653	DFA NCR CENTRAL	28/04/2018
18	RAICEA	FIRICA	14/11/2000	ROMANIAN	M	653526653	CONSTANTA	24/12/2015
19	STOIAN	VIOREL	15/2/1969	ROMANIAN	M	653438653	CONSTANTA	24/09/2015

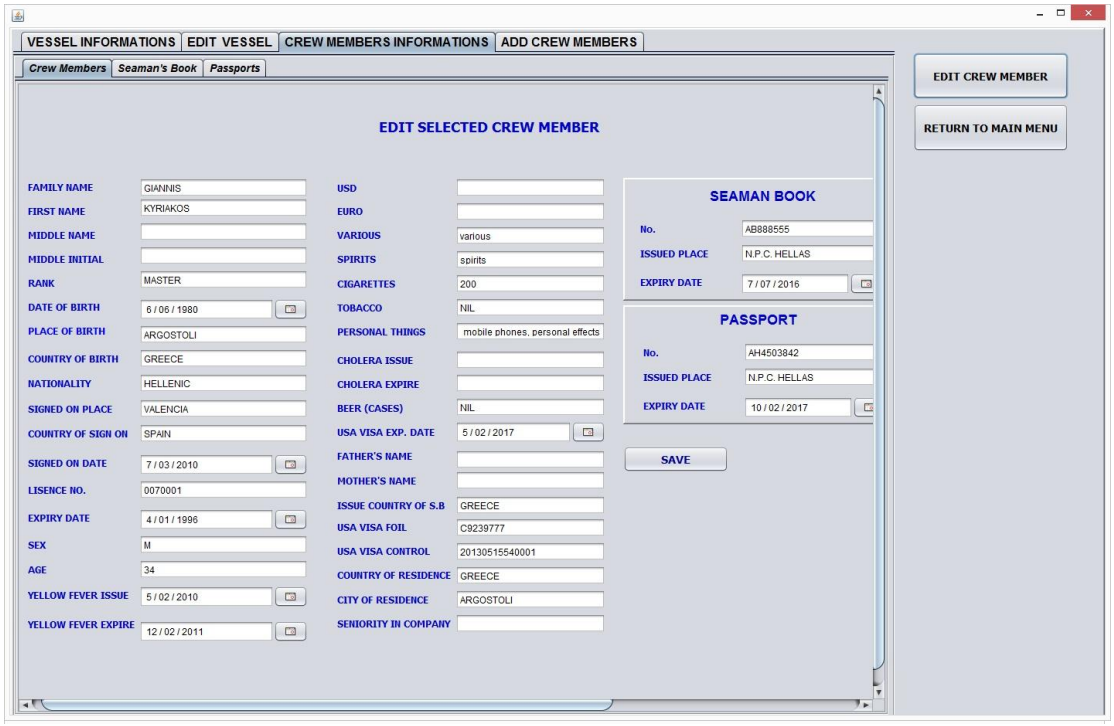
Εικόνα 4.11 Οθόνη προβολής πληροφοριών διαβατηρίων



No.	FAMILY NAME	FIRST NAME	MIDDLE NAME	MIDDLE INITIAL	RANK	DATE OF BIRTH	PLACE OF BIRTH	COUNTRY OF BIRTH	NATION
1	GIANNIS	KYRIAKOS			MASTER	06/06/1980	ARGOSTOLI	GREECE	HELLE
2	TUBO	RICARDO	DELA CRUZ	D	2ND OFF.	2/03/2012	CEBU CITY	PHILIPPINES	FILIPIN
3	TSOUPAKIS	ANDREAS			APP.OFF.	10/1/1993	MAROUSI	GREECE	HELLE
4	VLACHOPOULOS	ANASTASIOS			2ND ENG.	4/2/1964	PEIRAIAS	GREECE	HELLE
5	VASILAKIS	MICHAIL			3RD ENG.	9/5/1989	ATHINA	GREECE	HELLE
6	CHATZIOS	EVANGELOS			APP. ENG.	8/02/1980	FLORINA	GREECE	HELLE
7	VAMVAKOULAS	PANAGIOTIS			APP.ENG.	17/1/1993	ATHINA	GREECE	HELLE
8	DRAGOMIR	LORENZO			ELECTR.	16/1/1957	PUCIOASA	ROMANIA	ROMAN
9	STA MARIA	ALEXANDER	DUMAPAY	D	BOSUN	14/10/1964	GEN NATIVIDAD NE	PHILIPPINES	FILIPIN
10	DE LARA	ESTELITO JR	TAMPILIC	T	A.B	28/11/1981	LUBANG OC MDO	PHILIPPINES	FILIPIN
11	PASCUA	ARVIN	ENDOZO	E	A.B	10/8/1983	CALACA BATANGAS	PHILIPPINES	FILIPIN
12	MONTERO	REY	CABIGAS	C	A.B	15/11/1975	STA MARIA ILS SR	PHILIPPINES	FILIPIN
13	ANTIQUIERA	ERNELL	GOMEZ	G	DECK BOY	18/5/1987	BACOLOD CITY	PHILIPPINES	FILIPIN
14	BRIZUELA	DANIEL	BERGANTIN	B	OILER	17/11/1975	PILI CAMARINES S...	PHILIPPINES	FILIPIN
15	DEL MUNDO	GERARDO	BULAWAN	B	OILER	18/9/1968	CALBAYOG CITY	PHILIPPINES	FILIPIN
16	LAURENTE	WINNIE	BERMUDO	B	OILER	19/3/1980	SIAY ZAM DS	PHILIPPINES	FILIPIN
17	BUMACOD	VIRGILIO	BANTOLINO	B	WIPER	2/1/1969	MABINI	PHILIPPINES	FILIPIN
18	RAICEA	FIRICA			FITTER	14/11/2000	CONSTANTA	ROMANIA	ROMAN
19	STOIAN	VIOREL			FITTER	15/2/1969	CONSTANTA	ROMANIA	ROMAN

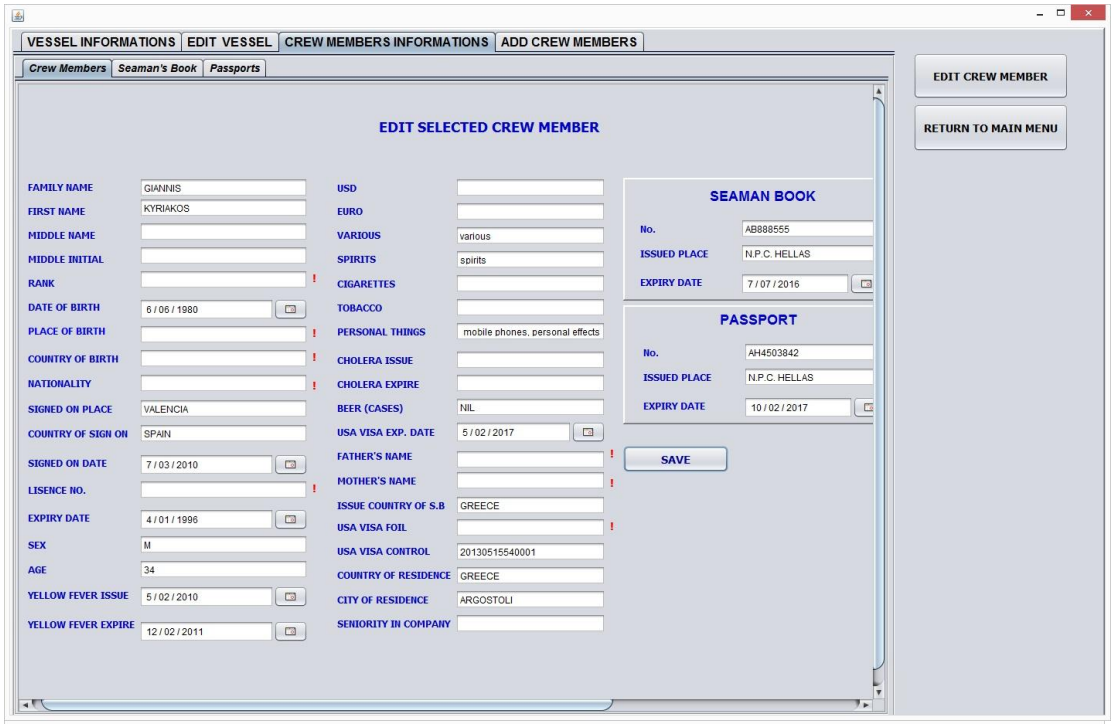
Εικόνα 4.12 Οθόνη προβολής επιλογή μέλους για επεξεργασία

Επιλέγοντας την υπό καρτέλα Crew Member's, εκτός απο την δυνατότητα που δίνεται στον χρήστη να δει τις πληροφορίες των μελών του πληρώματος, του δίνεται η δυνατότητα επεξεργασίας και διαγραφής των στοιχείων τους, επιλέγοντας ένα μέλος από τον πίνακα που τα απεικονίζει. Στη συνέχεια, επιλέγοντας το πλήκτρο Edit Crew member ο χρήστης μεταφέρεται στο πάνελ επεξεργασίας και τα στοιχεία του επιλεγμένου μέλους συμπληρώνονται στα αντίστοιχα πεδία επεξεργασίας.

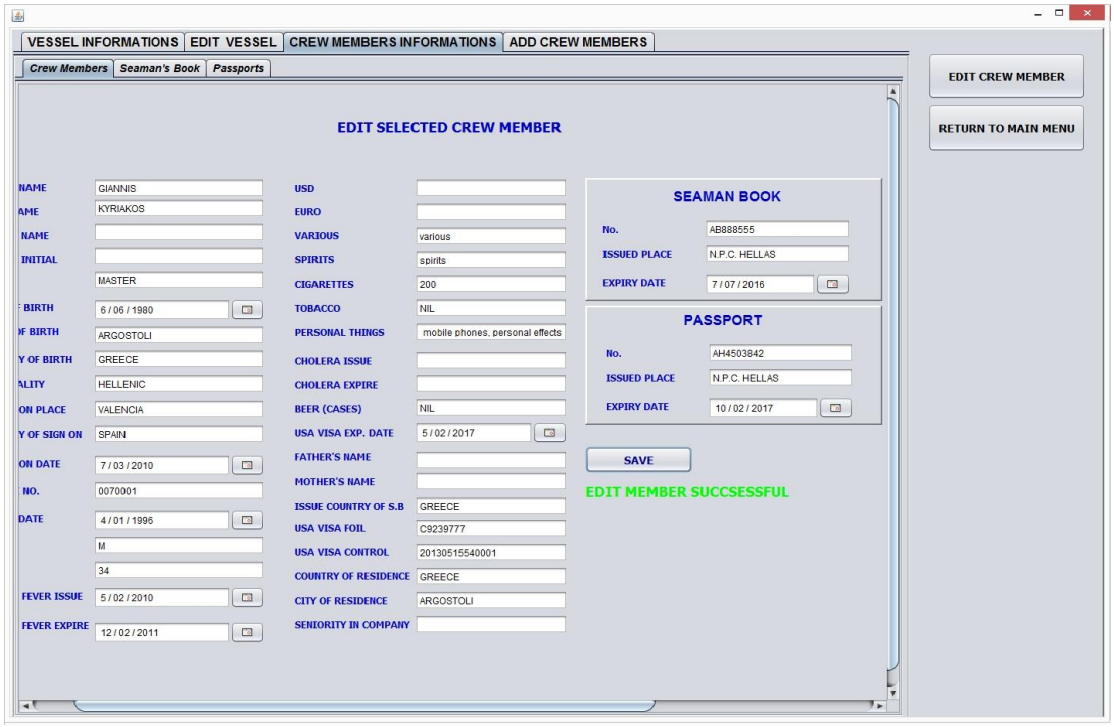


Εικόνα 4.13 Οθόνη προβολής επεξεργασία δεδομένων του επιλεγμένου μέλους

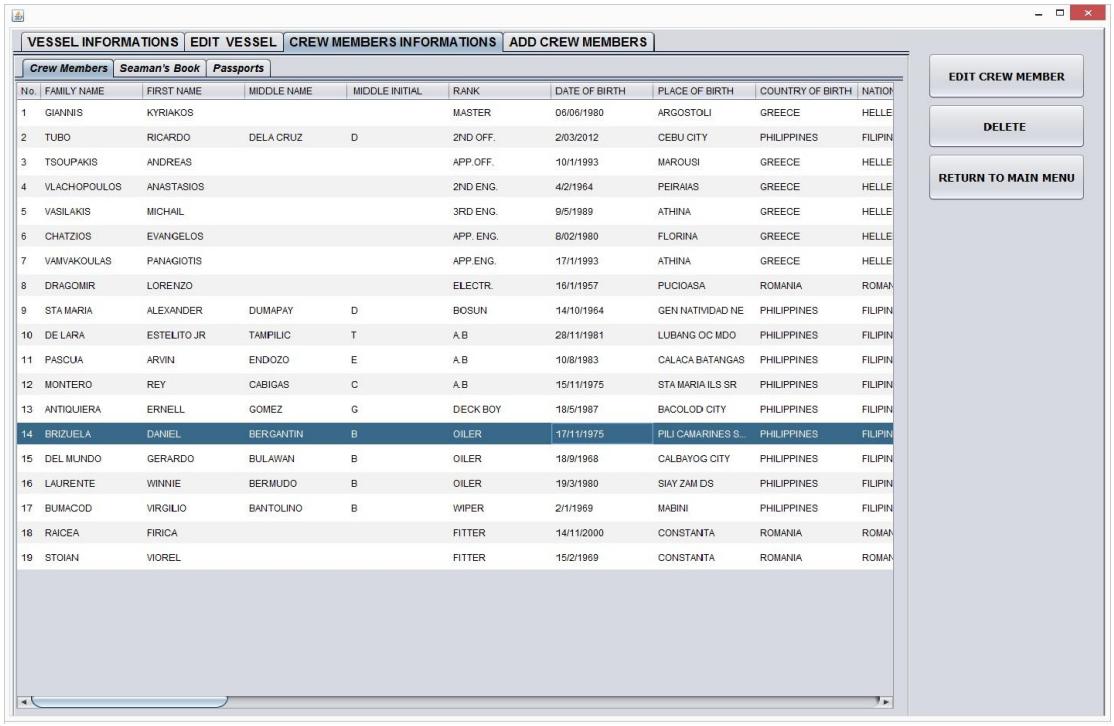
Στο σημείο αυτό ο χρήστης επεξεργάζεται τα στοιχεία του επιλεγμένου μέλους. Στη συνέχεια πιέζοντας το κουμπί Save γίνονται οι απαραίτητοι έλεγχοι για την εξασφάλιση της συμπλήρωσης των υποχρεωτικών πεδίων. Αν υπάρχουν κενά υποχρεωτικά πεδία εμφανίζονται μηνύματα λάθους για τα αντίστοιχα πεδία (εικόνα 4.14). Αν τα υποχρεωτικά πεδία έχουν συμπληρωθεί τα στοιχεία του μέλους που επεξεργάστηκε ο χρήστης αποθηκεύονται στην βάση δεδομένων, εμφανίζεται το κατάλληλο μήνυμα επιτυχίας και στην συνέχεια οι πίνακες απεικόνισης πληροφοριών της εφαρμογής ανανεώνονται (εικόνα 4.15).



Εικόνα 4.14 Οθόνη προβολής επεξεργασία δεδομένων του επιλεγμένου μέλους, εμφάνιση μηνυμάτων λάθους για παράληψη συμπλήρωσης υποχρεωτικών πεδίων



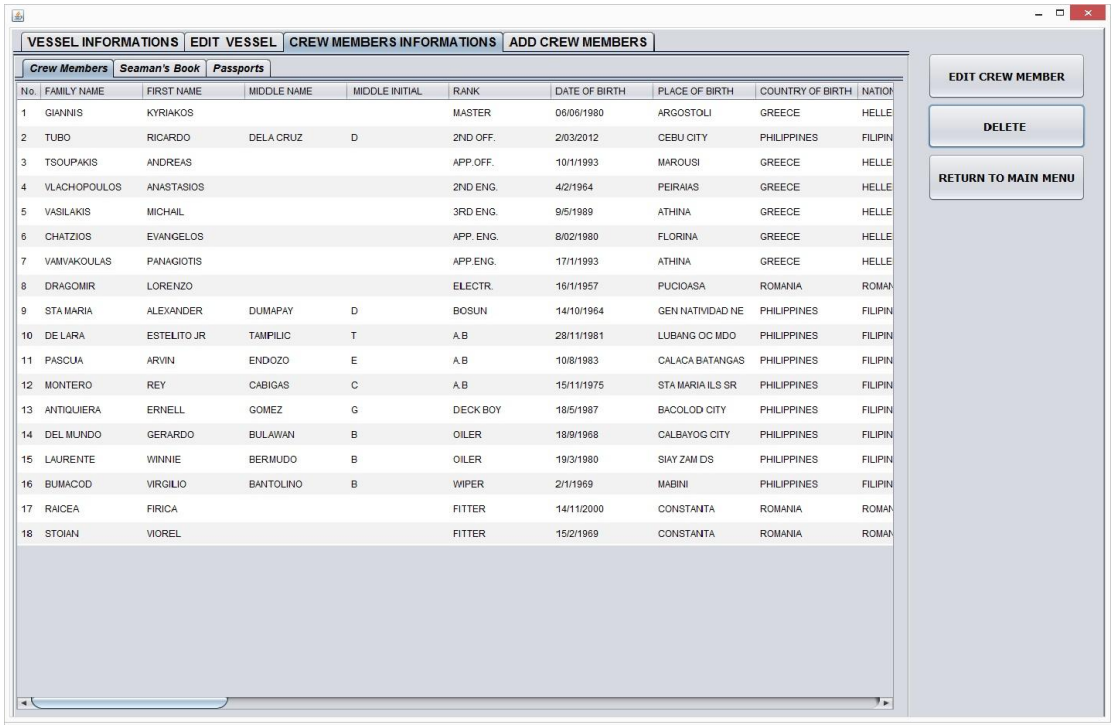
Εικόνα 4.15 Οθόνη προβολής επεξεργασία δεδομένων του επιλεγμένου μέλους, επιτυχημένη καταχώριση



No.	FAMILY NAME	FIRST NAME	MIDDLE NAME	MIDDLE INITIAL	RANK	DATE OF BIRTH	PLACE OF BIRTH	COUNTRY OF BIRTH	NATION
1	GIANNIS	KYRIAKOS			MASTER	06/06/1980	ARGOSTOLI	GREECE	HELLE
2	TUBO	RICARDO	DELA CRUZ	D	2ND OFF.	2/03/2012	CEBU CITY	PHILIPPINES	FILIPIN
3	TSOUPAKIS	ANDREAS			APP.OFF.	10/1/1993	MAROUSI	GREECE	HELLE
4	VLACHOPOULOS	ANASTASIOS			2ND ENG.	4/2/1964	PEIRAIAS	GREECE	HELLE
5	VASILAKIS	MICHAIL			3RD ENG.	9/5/1989	ATHINA	GREECE	HELLE
6	CHATZIOS	EVANGELOS			APP. ENG.	8/02/1980	FLORINA	GREECE	HELLE
7	VAMVAKOULAS	PANAGIOTIS			APP.ENG.	17/1/1993	ATHINA	GREECE	HELLE
8	DRAGOMIR	LORENZO			ELECTR.	16/1/1957	PUCIOASA	ROMANIA	ROMAN
9	STA MARIA	ALEXANDER	DUMAPAY	D	BOSUN	14/10/1964	GEN NATIVIDAD NE	PHILIPPINES	FILIPIN
10	DE LARA	ESTELITO JR	TAMPILIC	T	A.B	28/11/1981	LUBANG OC MDO	PHILIPPINES	FILIPIN
11	PASCUA	ARVIN	ENDOZO	E	A.B	10/8/1983	CALACA BATANGAS	PHILIPPINES	FILIPIN
12	MONTERO	REY	CABIGAS	C	A.B	15/11/1975	STA MARIA ILS SR	PHILIPPINES	FILIPIN
13	ANTIQUIERA	ERNELL	GOMEZ	G	DECK BOY	18/5/1987	BACOLOD CITY	PHILIPPINES	FILIPIN
14	BRIZUELA	DANIEL	BERGANTIN	B	OILER	17/11/1975	PILI CAMARINES S...	PHILIPPINES	FILIPIN
15	DEL MUNDO	GERARDO	BULAWAN	B	OILER	18/9/1968	CALBAYOG CITY	PHILIPPINES	FILIPIN
16	LAURENTE	WINNIE	BERMUDO	B	OILER	19/3/1980	SIAY ZAM DS	PHILIPPINES	FILIPIN
17	BUMACOD	VIRGILIO	BANTOLINO	B	WIPER	2/1/1969	MABINI	PHILIPPINES	FILIPIN
18	RAICEA	FIRICA			FITTER	14/11/2000	CONSTANTA	ROMANIA	ROMAN
19	STOIAN	VIOREL			FITTER	15/2/1969	CONSTANTA	ROMANIA	ROMAN

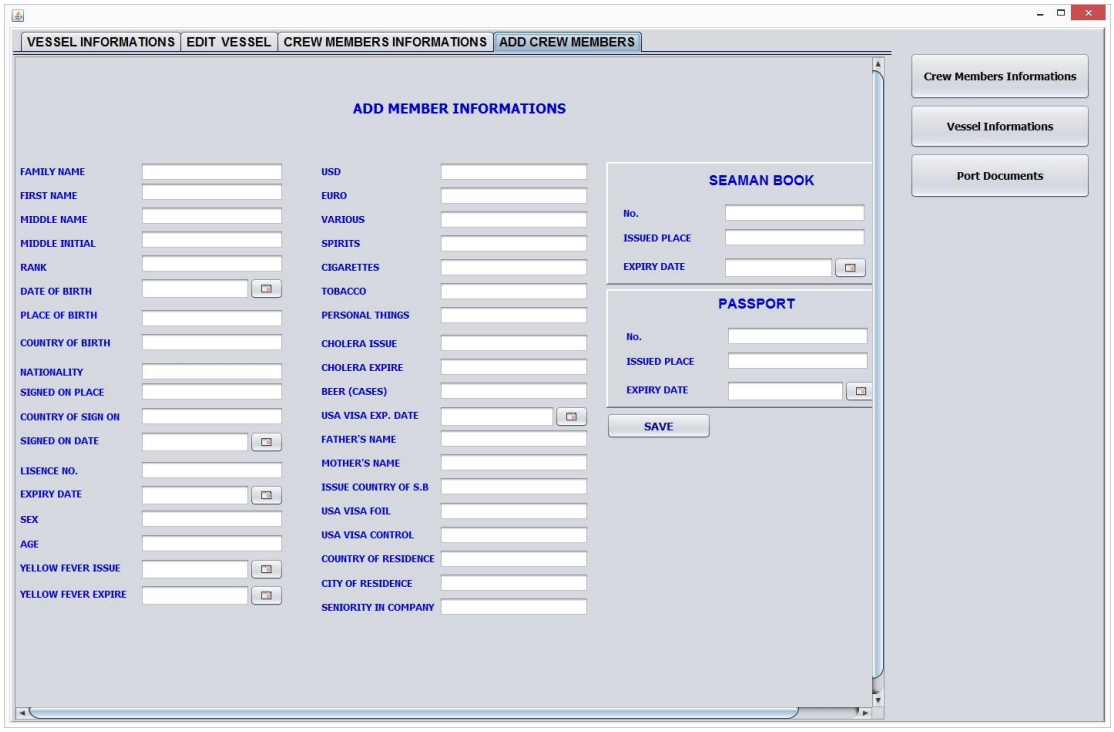
Εικόνα 4.16 Οθόνη προβολής επιλογή μέλους για διαγραφή

Επιλέγοντας κάποιο μέλος και επιλέγοντας το κουμπί Delete το επιλεγμένο μέλος του πληρώματος διαγράφεται από την βάση δεδομένων και στην συνέχεια οι πινάκες απεικόνισης ανανεώνονται με τα καινούργια δεδομένα εικόνα 4.17.



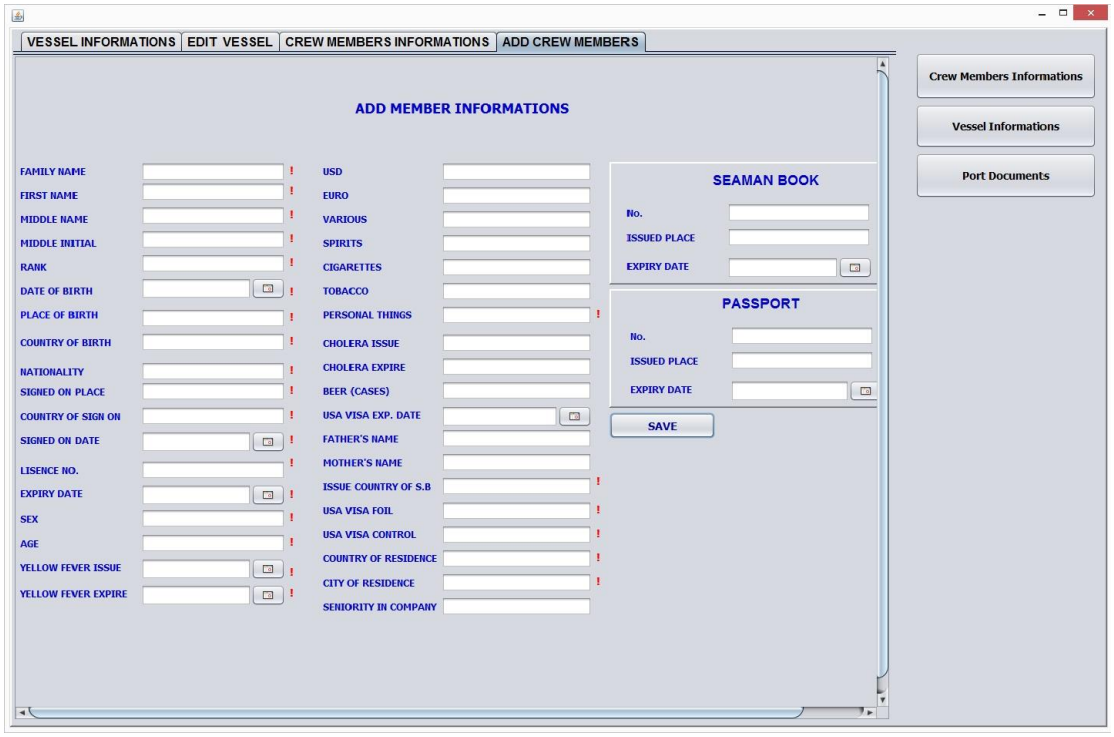
No.	FAMILY NAME	FIRST NAME	MIDDLE NAME	MIDDLE INITIAL	RANK	DATE OF BIRTH	PLACE OF BIRTH	COUNTRY OF BIRTH	NATION
1	GIANNIS	KYRIAKOS			MASTER	06/06/1980	ARGOSTOLI	GREECE	HELLE
2	TUBO	RICARDO	DELA CRUZ	D	2ND OFF.	2/03/2012	CEBU CITY	PHILIPPINES	FILIPIN
3	TSOUPAKIS	ANDREAS			APP.OFF.	10/1/1993	MAROUSI	GREECE	HELLE
4	VLACHOPOULOS	ANASTASIOS			2ND ENG.	4/2/1964	PEIRAIAS	GREECE	HELLE
5	VASILAKIS	MICHAIL			3RD ENG.	9/5/1989	ATHINA	GREECE	HELLE
6	CHATZIOS	EVANGELOS			APP. ENG.	8/02/1980	FLORINA	GREECE	HELLE
7	VAMVAKOULAS	PANAGIOTIS			APP.ENG.	17/1/1993	ATHINA	GREECE	HELLE
8	DRAGOMIR	LORENZO			ELECTR.	16/1/1957	PUCIOASA	ROMANIA	ROMAN
9	STA MARIA	ALEXANDER	DUMAPAY	D	BOSUN	14/10/1964	GEN NATIVIDAD NE	PHILIPPINES	FILIPIN
10	DE LARA	ESTELITO JR	TAMPILIC	T	A.B	28/11/1981	LUBANG OC MDO	PHILIPPINES	FILIPIN
11	PASCUA	ARVIN	ENDOZO	E	A.B	10/8/1983	CALACA BATANGAS	PHILIPPINES	FILIPIN
12	MONTERO	REY	CABIGAS	C	A.B	15/11/1975	STA MARIA ILS SR	PHILIPPINES	FILIPIN
13	ANTIQUIERA	ERNELL	GOMEZ	G	DECK BOY	18/5/1987	BACOLOD CITY	PHILIPPINES	FILIPIN
14	DEL MUNDO	GERARDO	BULAWAN	B	OILER	18/9/1968	CALBAYOG CITY	PHILIPPINES	FILIPIN
15	LAURENTE	WINNIE	BERMUDO	B	OILER	19/3/1980	SIAY ZAM DS	PHILIPPINES	FILIPIN
16	BUMACOD	VIRGILIO	BANTOLINO	B	WIPER	2/1/1969	MABINI	PHILIPPINES	FILIPIN
17	RAICEA	FIRICA			FITTER	14/11/2000	CONSTANTA	ROMANIA	ROMAN
18	STOIAN	VIOREL			FITTER	15/2/1969	CONSTANTA	ROMANIA	ROMAN

Εικόνα 4.17 Οθόνη προβολής ο πίνακας μετά την διαγραφή του επιλεγμένου μέλους

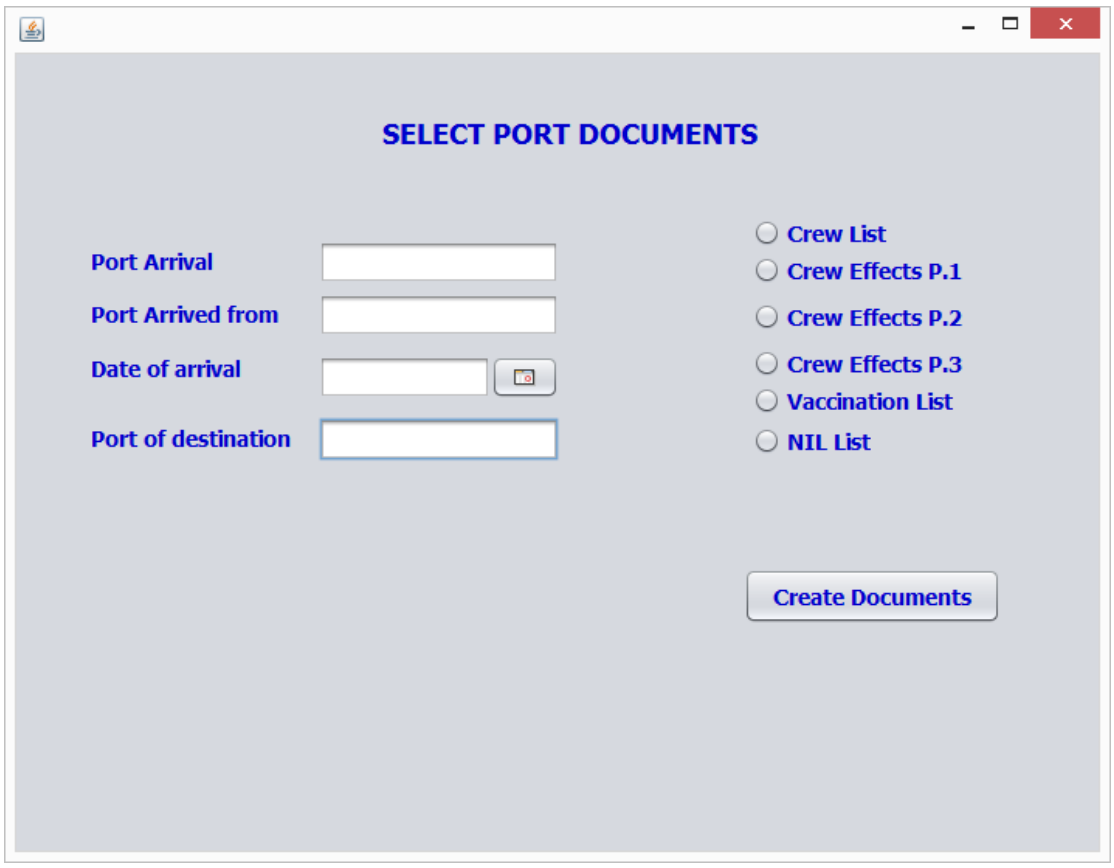


Εικόνα 4.18 Οθόνη προβολής φόρμα συμπλήρωσης για προσθήκη νέων μελών στο πλήρωμα

Μια ακόμα λειτουργία που προσφέρει η εφαρμογή μας στον χρήστη είναι η προσθήκη ενός νέου μέλους στο πλήρωμα. Ο χρήστης συμπληρώνει όλα τα υποχρεωτικά πεδία, σε περίπτωση παράληψης κάποιου ενημερώνεται με μηνύματα λάθους εικόνα 4.19. Σε περίπτωση που το σύνολο των υποχρεωτικών πεδίων έχουν συμπληρωθεί τα στοιχεία του νέου μέλους αποθηκεύονται στην βάση δεδομένων και οι πίνακες προβολής πληροφοριών για τα μέλη του πληρώματος ανανεώνονται.



Εικόνα 4.19 Οθόνη προβολής φόρμα συμπλήρωσης για προσθήκη νέων μελών στο πλήρωμα, εμφάνιση μηνυμάτων λάθους για παράληψη συμπλήρωσης υποχρεωτικών πεδίων



Εικόνα 4.20 Οθόνη προβολής επιλογή εγγράφων xls για δημιουργία

Τέλος επιλέγοντας το κουμπί Port Documents εμφανίζεται νέο παράθυρο στον χρήστη όπου αφού συμπληρώσει όλα τα υποχρεωτικά πεδία έχει την δυνατότητα να επιλέξει τα έγγραφα xls που χρειάζεται και στην συνέχεια τα έγγραφα αυτά θα συμπληρωθούν αυτόματα.

Κεφάλαιο 5

Συμπεράσματα και Μελλοντικές επεκτάσεις

5.1 Συμπεράσματα

Στα πλαίσια της εργασίας αυτής πραγματοποιήθηκε μελέτη σχετικά με τα Συστήματα Διαχείρισης Στόλου και αναπτύχθηκε λογισμικό για την διαχείριση του πληρώματος πλοίων. Πιο συγκεκριμένα μελετήσαμε λειτουργίες και χαρακτηριστικά συστημάτων διαχείρισης. Επίσης αναφέρθηκαν συστήματα διαχείρισης στόλου σε πραγματικό χρόνο. Ακόμα εξετάστηκε η διαδικασία διαχείρισης και δρομολόγησης του στόλου. Συνοψίζοντας καταλήγουμε στο συμπέρασμα ότι ο όγκος των δεδομένων που έχουν να διαχειριστούν οι ναυτιλιακές εταιρείες, η πολυπλοκότητα των υπολογισμών και η διασπορά της πληροφορίας σε διάφορες πηγές, καθιστά αναγκαία την χρήση πληροφοριακών συστημάτων και του διαδικτύου για την διευκόλυνση της διαδικασίας διαχείρισης του στόλου τους.

Το λογισμικό που αναπτύχθηκε προσφέρει τη δυνατότητα στον χρήστη να διαχειριστεί στοιχεία του πλοίου και του πληρώματος. Επίσης, δίνει την δυνατότητα για αυτόματη δημιουργία εγγράφων που αφορούν τα στοιχεία του πληρώματος και του πλοίου. Τα έγγραφα αυτά είναι απαραίτητα για την άφιξη των πλοίων στα λιμάνια προορισμού τους. Η χρήση της εφαρμογής που δημιουργήθηκε μπορεί να φανεί χρήσιμη αφού προσφέρει βασικές και εύχρηστες λειτουργίες για την διαχείριση δεδομένων του πληρώματος και του πλοίου. Παράλληλα προστατεύει τον χρήστη από πιθανά λάθη καταχώρησης των δεδομένων και εκμηδενίζει τον χρόνο παραγωγής χρήσιμων εγγράφων εξασφαλίζοντας την ορθή δημιουργίας τους.

5.2 Μελλοντικές επεκτάσεις

Το σύστημα που αναπτύχθηκε στην εν λόγω εργασία είναι υπεύθυνο για την οργάνωση και διαχείριση δεδομένων του πληρώματος καθώς και την εξαγωγή εγγράφων βασισμένων στα δεδομένα αυτά. Η εφαρμογή μας μπορεί να έχει σενάρια επέκτασης. Συγκεκριμένα μπορεί να αναπτυχθεί λογισμικό το οποίο θα έχει ως σκοπό την αυτόματη αναγνωρίσει των εκάστοτε απαιτήσεων για τα λιμάνια προορισμού του πλοίου καθώς και την τακτοποίηση σχετικών υποχρεώσεων. Επίσης μπορεί να δημιουργηθεί σύστημα για την διαχείριση ρύπων του πλοίου αναλόγως των τοπικών απαιτήσεων. Ειδικότερα οι επιπτώσεις από τη θαλάσσια ρύπανση είναι σημαντικές σε όλα τα επίπεδα. Με την ρύπανση των θαλασσών πλήττονται τα οικοσυστήματα, η ανθρώπινη υγεία, η οικονομική δραστηριότητα και η ευζωία. Επομένως η ανάπτυξη ενός συστήματος εκπομπής ρύπων που θα λειτουργεί βάση των θαλάσσιων ζωνών και των τοπικών απαιτήσεων θα μπορούσε να είναι μια αρκετά χρήσιμη επέκταση.

Βιβλιογραφία

[1] Laporte G., Crainic, T.G. (Eds.) (2000), Fleet Management & Logistics, Kluwer, Boston, US

[2]Gruhn, V., Hulder, M., Ijioui, R. and Schope, L. (2003), Mobile Communication Systems for Truckage Companies. In Giaglis, G.M., Werthner, H., Tschammer, V. and Froeschl, K.A. (Eds.), Proceedings of the 2nd International Conference on Mobile Business, Vienna, Austria, 23-24 June, pp. 337-344.

[3]Rushton A., Oxley, J., Croucher P. (2000), "The Handbook of Logistics and Distribution Management", 2nd Edition, © The Institute of Logistics and Transport, UK

[4]Psaraftis, H. N. (1995) "Dynamic Vehicle Routing: Status and Prospects", Annals of Operations Research 611, pp. 143-164

[5] Campbell, A., Clarke, L., Kleywergt, A., Savelsbergh, M. (1998) "The inventory routing problem", In Laporte G., Crainic, T.G. (Eds.) Fleet Management & Logistics, Kluwer, Boston, US

[6] Ichoua, S., Gendreau, M., Potvin, J.Y. (2003) "Vehicle dispatching with time-dependent travel times", European Journal of Operational Research 144, pp.379-396

[7] Kim, S, Lewis, M.E., White C.C. (2003) "Optimal Vehicle Routing with Real-Time Information", Working Paper, University of Michigan

[8]Fleischmann, B., Gnutzmann, S. Sandvoss, E. (2004b), "Dynamic vehicle routing based on on-line traffic information", Transportation Science Vol. 38, No. 4 pp.420-433

[9]Taniguchi, E., Shimamoto, H. (2004) "Intelligent transportation system based dynamic vehicle routing and scheduling with variable travel times" Transportation research Part C, Vol. 12, pp.235-250

[10]Haghani, A., Jung S. (2004), "A dynamic vehicle routing problem with time-dependent travel times", Computers & Operations Research, (In press)

[11] DYNALOG (Dynamic logistics for distribution networks), ESPRIT[35] Program 26387, http://www.cordis.lu/esprit/src/26387.htm.

[12]Savelsbergh M.W.P., Local Search for Routing Problems with Time Windows, Annals of Operations Research 4, 1985, pp. 285-305.

[13] Holland J. H., Adaptation in Natural and Artificial systems, University of Michigan Press, Ann Arbor, 1975

[14] K. Fagerholt and M. Christiansen. A travelling salesman problem with allocation, time window and precedence constraints – an application to ship scheduling. International Transactions in Operational Research,

[15]G. BrØnmo, M. Christiansenb, K. Fagerholt, B. Nygreen. A multi-start local search heuristic for ship scheduling— a computational study. Computers & Operations Research 34 (2007)

[16] Burmester & Vogel: www.voyage-estimation.com

[17] Danaos: www.danaos.com

[18] Baltic exchange: www.balticexchange.com

[19] Intertanko: www.intertanko.com

[20] Lloyds: www.lloydsmiu.com/lmiu/index.htm

[21] BRL Shipping Consultants: www.brldata.com

[22] Regan, A.C., Herrmann, J., Lu, X. (2002) "The relative performance of heuristics for the dynamic travelling salesman problem", Proceedings of the 81st Annual Meeting of the Transportation Research Board, Washigton, DC

[23] Zhang, H. M. (2000) "Recursive prediction of traffic conditions with neural networks", Journal of Transportation Engineering, 126(6), pp. 472–481

[24] Whitlock, M. E. and Queen, C. M. (2000) "Modelling a traffic network with missing data", Journal of Forecasting, 19, pp. 561–574

[25] Van Arem, B., Van der Vlist, M. J. M., Muste, M. and Smulders, S. A. (1997) "Travel time estimation in the Gerdien project", International Journal of Forecasting, 13, pp. 73–85

[26] Yin, R.K. (1984) Case Study Research: Design and Methods Sage Publications, Thousand Oaks, CA

[27] Levinson, H. S. and Lomax, T. J. (1996) "Developing a travel time congestion index", Transportation Research Record, 1564, pp. 1–10

[28] Law, A., and Kelton D. (1991) "Simulation Modelling and Analysis", McGraw-Hill, New York, US

[29] Lan, C.-J. and Miaou, S.-P. (1999) "Real-time prediction of traffic flows using dynamic generalized linear models", Transportation Research Record, 1678, pp. 168–178

Παράρτημα

Κώδικας υλοποίησης σημαντικότερων κλάσεων

Πακέτο CrewDataDBcons κλάση db_connection

```
package CrewDataDBcons;

import javaapplication5.*;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javaapplication5.CrewMember;



public class db_connection {

    Connection con;

    public void connect() {
        try {
            String url =
"jdbc:mysql://galaxy.hua.gr:3306/it****?user=it****&password=****
**";
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            con = DriverManager.getConnection(url);
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
" EX1", ex);
        } catch (ClassNotFoundException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
" EX2", ex);
        } catch (Exception e) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, e);
        }

    }

    public void closeConnection() {
        try {
            this.con.close();
```

```
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (Exception e) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, e);
        }
    }

    public ArrayList<CrewMember> getCrewMembers() {
        ArrayList<CrewMember> CrewMembersList = new
ArrayList<CrewMember>();

        String familyName = "";
        String fistName = "";
        String middleName = "";
        String middleInitial = "";
        String rank = "";
        String dateOFBirth = "";
        String placeOfBirth = "";
        String countryOfBirth = "";
        String nationality = "";
        String signedOnPlace = "";
        String countryOfSignOn = "";
        String signOnDate = "";
        String lisenceNo = "";
        String expiryDate = "";
        String sex = "";
        String age = "";
        String yellowFeverIssue = "";
        String yellowFeverExpire = "";
        String usd = "";
        String euro = "";
        String various = "";
        String spirits = "";
        String cigarettes = "";
        String tobacco = "";
        String personalThings = "";
        String choleraIssue = "";
        String choleraExpire = "";
        String beerCases = "";
        String usaVisaExpDate = "";
        String fatherName = "";
        String motherName = "";
        String usaVisaFoil = "";
        String issueCountryOfSBforSanCrewL = "";
        String usaVisaControl = "";
        String countryOfResidence = "";
        String cityOfResidence = "";
        String seniorityInCompany = "";

        int counter = 0;

        Statement stmt = null;
        ResultSet rs = null;
```

```
        try {
            stmt = this.con.createStatement();
            String tempStmt = "SELECT * FROM it20826.CrewList";
            rs = stmt.executeQuery(tempStmt);
            while (rs.next()) {

                // tempArray = new ArrayList<String>();
                familyName = rs.getString("familyName");
                fistName = rs.getString("firstName");
                middleName = rs.getString("middleName");
                middleInitial = rs.getString("middleInitial");
                rank = rs.getString("rank");
                dateOFBirth = rs.getString("dateOfBirth");
                placeOfBirth = rs.getString("placeOfBirth");
                countryOfBirth = rs.getString("countryOfBirth");
                nationality = rs.getString("nationality");
                signedOnPlace = rs.getString("signedOnPlace");
                countryOfSignOn =
rs.getString("countryOfSignOn");
                signOnDate = rs.getString("signOnDate");
                lisenceNo = rs.getString("lisenceNo");
                expiryDate = rs.getString("expiryDate");
                sex = rs.getString("sex");
                age = rs.getString("age");
                yellowFeverIssue =
rs.getString("yellowFeverIssue");
                yellowFeverExpire =
rs.getString("yellowFeverExpire");
                usd = rs.getString("usd");
                euro = rs.getString("euro");
                various = rs.getString("various");
                spirits = rs.getString("spirits");
                cigarettes = rs.getString("cigarettes");
                tobacco = rs.getString("tobacco");
                personalThings = rs.getString("personalThings");
                choleraIssue = rs.getString("choleraIssue");
                choleraExpire = rs.getString("choleraExpire");
                beerCases = rs.getString("beerCases");
                usaVisaExpDate = rs.getString("usaVisaExpDate");
                fatherName = rs.getString("fatherName");
                motherName = rs.getString("motherName");
                usaVisaFoil = rs.getString("usaVisaFoil");
                issueCountryOfSBforSanCrewL =
rs.getString("issueCountryOfSBfotSanCrewL");
                usaVisaControl = rs.getString("usaVisaControl");
                countryOfResidence =
rs.getString("countryOfResidence");
                cityOfResidence =
rs.getString("cityOfResidence");
                seniorityInCompany =
rs.getString("seniorityInCompany");
                CrewMember tempCrewMembers = new
CrewMember(familyName, fistName, middleName, middleInitial, rank,
dateOFBirth, placeOfBirth, countryOfBirth, nationality,
signedOnPlace, countryOfSignOn, signOnDate, lisenceNo,
expiryDate, sex, age, yellowFeverIssue, yellowFeverExpire, usd,
euro, various, spirits, cigarettes, tobacco, personalThings,
```

```
choleraIssue, choleraExpire, beerCases, usaVisaExpDate,
fatherName, motherName, usaVisaFoil, issueCountryOfSBforSanCrewL,
usaVisaControl, countryOfResidence, cityOfResidence,
seniorityInCompany);

                CrewMembersList.add(tempCrewMembers);

                counter++;
                System.out.println(counter);

            }

        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return CrewMembersList;
    }

    public ArrayList<SeamansBook> getSeamansBook() {
        ArrayList<SeamansBook> SeamansBookList = new
ArrayList<SeamansBook>();

        String NoSea;
        String issuebPlaceSea;
        String expiryDateSea;
        String visa;


        int counter = 0;

        Statement stmt1 = null;
        ResultSet rs = null;
        try {
            stmt1 = this.con.createStatement();
            String tempStmt1 = "SELECT * FROM  SemansBook";
            rs = stmt1.executeQuery(tempStmt1);
            while (rs.next()) {

                NoSea = rs.getString("No");
                issuebPlaceSea = rs.getString("IssuedPlace");
                expiryDateSea = rs.getString("ExpiryDate");
                visa=rs.getString("UsaVisaFoil");


                SeamansBook tempSeamansBook = new
SeamansBook(NoSea, issuebPlaceSea, expiryDateSea);

                SeamansBookList.add(tempSeamansBook);

                counter++;
                System.out.println(counter);

            }
```

```
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return SeamansBookList;
    }

    public ArrayList<Passport> getPassport() {
        ArrayList<Passport> PassportList = new
ArrayList<Passport>();

        String No;
        String issuedPlacePass;
        String expiryDatePass;

        int counter = 0;

        Statement stmt = null;
        ResultSet rs = null;
        try {
            stmt = this.con.createStatement();
            String tempStmt = "SELECT * FROM it20826.passport";
            rs = stmt.executeQuery(tempStmt);
            while (rs.next()) {

                No = rs.getString("No");
                issuedPlacePass = rs.getString("IssuedPlace");
                expiryDatePass = rs.getString("ExpiryDate");
                Passport tempPassport = new Passport(No,
issuedPlacePass, expiryDatePass);

                PassportList.add(tempPassport);

                counter++;
                System.out.println(counter);

            }

        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return PassportList;
    }


     public boolean addCrewMemberData(CrewMember crewMem) {
        boolean success = true;




        try {
```

```
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("Insert into CrewList
values('" + crewMem.getFamilyName() + "','" +
crewMem.getFistName() + "','"
                    + crewMem.getMiddleName() + "','" +
crewMem.getMiddleInitial()+ "','" + crewMem.getRank() + "','"
+crewMem.getDaeOFBirth()
                    + "','" + crewMem.getPlaceOfBirth() + "','"
+crewMem.getCountryOfBirth() + "','" + crewMem.getNationality()
                    + "','" + crewMem.getSignedOnPlace() + "','"
+ crewMem.getCountryOfSignOn()+ "','" + crewMem.getSignOnDate() +
"','" + crewMem.getLisenceNo()
                    + "','" + crewMem.getExpiryDate() + "','" +
crewMem.getSex()+ "','" + crewMem.getAge()+ "','" +
crewMem.getYellowFeverIssue()
                    + "','" + crewMem.getYellowFeverExpire()+
"','" + crewMem.getUsd() + "','" + crewMem.getEuro()+ "','" +
crewMem.getVarious()
                    + "','" + crewMem.getSpirits()+ "','" +
crewMem.getCigarettes()+ "','" + crewMem.getTobacco()+ "','" +
crewMem.getPersonalThings()
                    + "','" + crewMem.getCholeraIssue()+ "','" +
crewMem.getCholeraExpire()+ "','" + crewMem.getBeerCases()+ "','"
+ crewMem.getUsaVisaExpDate()
                    + "','" + crewMem.getFatherName()+ "','" +
crewMem.getMotherName()+ "','"
+crewMem.getIssueCountryOfSBforSanCrewL() + "','" +
crewMem.getUsaVisaFoil()
                    + "','" + crewMem.getUsaVisaControl()+ "','"
+ crewMem.getCountryOfResidence()+ "','" +
crewMem.getCityOfResidence()+ "','" +
crewMem.getSeniorityInCompany()+ "')");
            success=true;
        } catch (SQLException ex) {
            success = false;

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }
        System.out.println(success);









        return success;
    }

public boolean addPassMemberData(CrewMember crewMem,Passport pass
){

     boolean success2 = true;
```

```
         try {
            Statement stmt2;
            stmt2 = this.con.createStatement();

            success2 = stmt2.execute("Insert into passport
values('" + pass.getNo()+ "','" + pass.getIssuedPlacePass()+
"','" + pass.getExpiryDatePass()+ "','" +
crewMem.getUsaVisaFoil()+  "')");
            success2=true;
        } catch (SQLException ex) {
            success2 = false;

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }


    return success2;
}

    public boolean addSeamansBookData(CrewMember
crewMem,SeamansBook seamans ){

     boolean success1 = true;

         try {
            Statement stmt1;
            stmt1 = this.con.createStatement();

            success1 = stmt1.execute("Insert into SemansBook
values('" + seamans.getNoSea() + "','" +
seamans.getIssuebPlaceSea()+ "','" + seamans.getExpiryDateSea()+
"','" + crewMem.getUsaVisaFoil()+ "')");
            success1=true;
        } catch (SQLException ex) {
            success1 = false;

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }


    return success1;

}


       public boolean deleteMember(String visaFoil){
       boolean success = false;
       String query = null;

        try {
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("Delete from  CrewList where
(usaVisaFoil='" + visaFoil +"')");
```

```
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

         try {
            Statement stmt1;
            stmt1 = this.con.createStatement();

            success = stmt1.execute("Delete from  SemansBook
where (UsaVisaFoil='" + visaFoil +"')");

        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

          try {
            Statement stmt2;
            stmt2 = this.con.createStatement();

            success = stmt2.execute("Delete from  passport where
(UsaVisaFoil='" +visaFoil+"')");

        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return success;
    }


public int checkLogin(String userName ,String pass){
        int user=-1;

       int result = -1;

        Statement stmt = null;
        ResultSet rs = null;
        try {
            stmt = this.con.createStatement();
            String tempStmt = "Select * FROM logIn where
(UserName='" + userName + "' and Password='" + pass + "')";
            rs = stmt.executeQuery(tempStmt);
               if(rs.next()){
                  user=1;
               }else{
                user=-1;
               }
        } catch (SQLException ex) {
```

```
Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        System.out.println(result+"LOG IN");
        return user; //epistrefei -1 an to login htan apotuxhmeno
kai ton typo xristi an htan petixhmeno admin/user









}

public ArrayList<Vessel> getVesselDB() {
        ArrayList<Vessel> VesselList = new ArrayList<Vessel>();

        String ShipsName = "";
        String CallSign = "";
        String ImoNumber = "";
        String OfficialNumber = "";
        String Nationality = "";
        String TypeOfVessel = "";
        String PortOfRegistry = "";
        String CountryOfRegistry = "";
        String GrossTonnage = "";
        String NetTonnage = "";
        String DeadWeight = "";
        String Owners = "";
        String Operators = "";
        String VoyageNumber = "";
        String MastersName = "";
        String SsoName = "";
        String Charterers = "";
        String PortOfArrival = "";
        String DateOfArrival = "";
        String PortArrivedFrom = "";
        String NextPort = "";
        String Arrival = "";
        String Departure = "";
        String NumberOfCrew = "";
        String CargoMT = "";
        String DateOfDeparture = "";
        String TimeOfArrival = "";
        String PeriodOfStay = "";
        String DateTimeOfArrivalDeparture = "";
        String OwnersRegisteredNo = "";
        String CompanyIdentificationNo = "";


        int counter = 0;

        Statement stmt = null;
        ResultSet rs = null;
```

```
        try {
            stmt = this.con.createStatement();
            String tempStmt = "SELECT * FROM Vessel ";
            rs = stmt.executeQuery(tempStmt);
            while (rs.next()) {

                ShipsName = rs.getString("ShipsName");
                CallSign = rs.getString("CallSign");
                ImoNumber = rs.getString("ImoNumber");
                OfficialNumber = rs.getString("OfficialNumber");
                Nationality = rs.getString("Nationality");
                TypeOfVessel = rs.getString("TypeOfVessel");
                PortOfRegistry = rs.getString("PortOfRegistry");
                CountryOfRegistry =
rs.getString("CountryOfRegistry");
                GrossTonnage = rs.getString("GrossTonnage");
                NetTonnage = rs.getString("NetTonnage");
                DeadWeight = rs.getString("DeadWeight");
                Owners = rs.getString("Owners");
                Operators = rs.getString("Operators");
                VoyageNumber = rs.getString("VoyageNumber");
                MastersName = rs.getString("MastersName");
                SsoName = rs.getString("SsoName");
                Charterers = rs.getString("Charterers");
                PortOfArrival = rs.getString("PortOfArrival");
                DateOfArrival = rs.getString("DateOfArrival");
                PortArrivedFrom =
rs.getString("PortArrivedFrom");
                NextPort = rs.getString("NextPort");
                Arrival = rs.getString("Arrival");
                Departure = rs.getString("Departure");
                NumberOfCrew = rs.getString("NumberOfCrew");
                CargoMT = rs.getString("CargoMT");
                DateOfDeparture =
rs.getString("DateOfDeparture");
                TimeOfArrival = rs.getString("TimeOfArrival");
                PeriodOfStay = rs.getString("PeriodOfStay");
                DateTimeOfArrivalDeparture =
rs.getString("DateTimeOfArrivalDeparture");
                OwnersRegisteredNo =
rs.getString("OwnersRegisteredNo");
                CompanyIdentificationNo =
rs.getString("CompanyIdentificationNo");

                Vessel tempVessel = new Vessel(ShipsName,
CallSign, ImoNumber, OfficialNumber, Nationality, TypeOfVessel,
PortOfRegistry, CountryOfRegistry, GrossTonnage, NetTonnage,
DeadWeight, Owners, Operators, VoyageNumber, MastersName,
SsoName, Charterers, PortOfArrival, DateOfArrival,
PortArrivedFrom, NextPort, Arrival, Departure, NumberOfCrew,
CargoMT,
                        DateOfDeparture, TimeOfArrival,
PeriodOfStay, DateTimeOfArrivalDeparture, OwnersRegisteredNo,
CompanyIdentificationNo);


                VesselList.add(tempVessel);
```

```
                counter++;
                System.out.println(counter);

            }

        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }


        return VesselList;
    }

 public boolean updateVesselData(Vessel vessel) {
        boolean success = false;

        try {
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("UPDATE Vessel SET
ShipsName='" + vessel.getShipsName() + "', CallSign='" +
vessel.getCallSign()
                    + "', ImoNumber='" + vessel.getImoNumber() +
"', OfficialNumber='" + vessel.getOfficialNumber() + "',
Nationality='" + vessel.getNationality()

                    + "', TypeOfVessel='" +
vessel.getTypeOfVessel() + "', PortOfRegistry='" +
vessel.getPortOfRegistry() + "', CountryOfRegistry='" +
vessel.getCountryOfRegistry()

                    + "', GrossTonnage='" +
vessel.getGrossTonnage() + "', NetTonnage='" +
vessel.getNetTonnage()+ "', DeadWeight='" +
vessel.getDeadWeight()

                    + "', Owners='" + vessel.getOwners() + "',
Operators='" + vessel.getOperators() + "', VoyageNumber='" +
vessel.getVoyageNumber()

                    + "', MastersName='" +
vessel.getMastersName() + "', SsoName='" + vessel.getSsoName()+
"', Charterers='" + vessel.getCharterers()

                    + "', TypeOfVessel='" +
vessel.getTypeOfVessel() + "', PortOfRegistry='" +
vessel.getPortOfRegistry() + "', CountryOfRegistry='" +
vessel.getCountryOfRegistry()

                    + "', NumberOfCrew='" +
vessel.getNumberOfCrew() + "', CargoMT='" + vessel.getCargoMT()+
"', OwnersRegisteredNo='" + vessel.getOwnersRegisteredNo()
```

```
                    + "', CompanyIdentificationNo='" +
vessel.getCompanyIdentificationNo()+"'");//"' where ..="+

            success=true;
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }

        return success;
    }




  public boolean updateCrewmemberData(CrewMember crewMem) {
        boolean success = false;

        try {
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("UPDATE CrewList SET
familyName='" + crewMem.getFamilyName() + "', firstName='" +
crewMem.getFistName()


                    + "', middleName='" + crewMem.getMiddleName()
+ "', middleInitial='" + crewMem.getMiddleInitial() + "', rank='"
+ crewMem.getRank()

                    + "', dateOfBirth='" +
crewMem.getDaeOFBirth() + "', placeOfBirth='" +
crewMem.getPlaceOfBirth() + "', countryOfBirth='" +
crewMem.getCountryOfBirth()

                    + "', nationality='" +
crewMem.getNationality()  + "', signedOnPlace='" +
crewMem.getSignedOnPlace() + "', countryOfSignOn='" +
crewMem.getCountryOfSignOn()

                    + "', signOnDate='" + crewMem.getSignOnDate()
+ "', lisenceNo='" + crewMem.getLisenceNo() + "', expiryDate='" +
crewMem.getExpiryDate()

                    + "', sex='" + crewMem.getSex() + "', age='"
+ crewMem.getAge()+ "', yellowFeverIssue='" +
crewMem.getYellowFeverIssue()

                    + "', yellowFeverExpire='" +
crewMem.getYellowFeverExpire() + "', usd='" + crewMem.getUsd() +
"', euro='" + crewMem.getEuro()

                    + "', various='" + crewMem.getVarious() + "',
spirits='" +crewMem.getSpirits()+ "', cigarettes='" +
crewMem.getCigarettes()
```

```
                     + "', tobacco='" + crewMem.getTobacco() +
"', personalThings='" + crewMem.getPersonalThings() + "',
choleraIssue='" + crewMem.getCholeraIssue()

                    + "', choleraExpire='" +
crewMem.getCholeraExpire() + "', beerCases='" +
crewMem.getBeerCases()+ "', usaVisaExpDate='" +
crewMem.getUsaVisaExpDate()


                     + "', fatherName='" +
crewMem.getFatherName() + "', motherName='" +
crewMem.getMotherName() + "', issueCountryOfSBfotSanCrewL='" +
crewMem.getIssueCountryOfSBforSanCrewL()

                    + "', usaVisaControl='" +
crewMem.getUsaVisaControl() + "', countryOfResidence='" +
crewMem.getCountryOfResidence()+ "', cityOfResidence='" +
crewMem.getCityOfResidence()


                     + "', seniorityInCompany='" +
crewMem.getSeniorityInCompany() + "' WHERE usaVisaFoil='" +
crewMem.getUsaVisaFoil()+"'");

            success=true;
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }



        return success;
    }


  public boolean updateSeamansBookData(SeamansBook seamBook) {

   boolean success = false;


         try {
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("UPDATE SemansBook SET No='" +
seamBook.getNoSea().toString() + "', IssuedPlace='" +
seamBook.getIssuebPlaceSea() + "', ExpiryDate='" +
seamBook.getExpiryDateSea()
                    + "' WHERE UsaVisaFoil='"
+seamBook.getVisaFoilSea()+"'");

           success=true;
```

```
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }
        return success;



  }


  public boolean updatePassportData( Passport passport) {

   boolean success = false;


         try {
            Statement stmt;
            stmt = this.con.createStatement();

            success = stmt.execute("UPDATE passport SET No='" +
passport.getNo()+ "', IssuedPlace='" +
passport.getIssuedPlacePass() + "', ExpiryDate='" +
passport.getExpiryDatePass()
                    + "' WHERE UsaVisaFoil='"
+passport.getVisaFoil()+"'");

           success=true;
        } catch (SQLException ex) {

Logger.getLogger(db_connection.class.getName()).log(Level.SEVERE,
null, ex);
        }



        return success;
    }




}
```

Πακέτο CrewDataDBcons κλάση CrewDataDBLists

```
package CrewDataDBcons;
```

```
import java.util.ArrayList;
import javaapplication5.*;


public class CrewDataDBLists {



    public ArrayList<CrewMember> GetCrewMembers() {

        db_connection con = new db_connection();
        con.connect();
        ArrayList<CrewMember> CrewMembersListDB = new
ArrayList();
        CrewMembersListDB = con.getCrewMembers();;
        con.closeConnection();
        return CrewMembersListDB;
    }

    public ArrayList<SeamansBook> GetSeamansBook() {
        db_connection con = new db_connection();
        con.connect();
        ArrayList<SeamansBook> SeamanBookList = new ArrayList();
        ArrayList<CrewMember> CrewList = new ArrayList();

        SeamanBookList = con.getSeamansBook();
        CrewList = con.getCrewMembers();

        ArrayList<CrewMember> crewsea = new ArrayList();
        ArrayList<SeamansBook> SeamanJtableList = new
ArrayList();

        for (int i = 0; i < CrewList.size(); i++) {

            CrewMember crewMemSea = new
CrewMember(CrewList.get(i).familyName, CrewList.get(i).fistName,
CrewList.get(i).dateOFBirth, CrewList.get(i).placeOfBirth,
CrewList.get(i).fatherName,CrewList.get(i).motherName,
CrewList.get(i).usaVisaFoil);
            crewsea.add(crewMemSea);

        }
        int y = SeamanBookList.size();

        for (int i = 0; i < y; i++) {

            //CrewMember CrewMem= crewsea.get(i).;
            SeamansBook Seman = new
SeamansBook(SeamanBookList.get(i).NoSea,
SeamanBookList.get(i).issuebPlaceSea,
SeamanBookList.get(i).expiryDateSea, crewsea.get(i));
            SeamanJtableList.add(Seman);

        }

        // Passport pas= new Passport();
        con.closeConnection();
```

```
        return SeamanJtableList;
    }

    public ArrayList<Passport> GetPassport() {
        db_connection con = new db_connection();
        con.connect();
        ArrayList<Passport> PassportList = new ArrayList();
        ArrayList<CrewMember> CrewList = new ArrayList();

        PassportList = con.getPassport();
        CrewList = con.getCrewMembers();

        ArrayList<CrewMember> crewPass = new ArrayList();
        ArrayList<Passport> PassportJtableList = new ArrayList();

        for (int i = 0; i < CrewList.size(); i++) {

            CrewMember crewMemPass = new
CrewMember(CrewList.get(i).familyName, CrewList.get(i).fistName,
CrewList.get(i).dateOFBirth,
CrewList.get(i).nationality,CrewList.get(i).sex,
CrewList.get(i).usaVisaFoil);
            crewPass.add(crewMemPass);
        }

        for (int i = 0; i < PassportList.size(); i++) {

            Passport Pass = new Passport(PassportList.get(i).No,
PassportList.get(i).issuedPlacePass,
PassportList.get(i).expiryDatePass, crewPass.get(i));
            PassportJtableList.add(Pass);

        }

        // Passport pas= new Passport();
        con.closeConnection();
        return PassportJtableList;
    }

}
```

Πακέτο CrewDataDBcons κλάση TableModelClass

```
package CrewDataDBcons;

import java.util.ArrayList;
import javaapplication5.*;
import javax.swing.JTable;
import javax.swing.table.AbstractTableModel;
```

```
public class TableModelClass extends AbstractTableModel {


    private  String [] columnNames= {"No.","FAMILY NAME", "FIRST
NAME" , "MIDDLE NAME"," MIDDLE INITIAL","RANK","DATE OF
BIRTH","PLACE OF BIRTH","COUNTRY OF BIRTH","NATIONALITY","SIGNED
ON PLACE",
        "COUNTRY OF SIGN ON","SIGNED ON DATE","LISENCE
NO.","EXPIRY DATE","SEX","AGE","YELLOW FEVER ISSUE","YELLOW FEVER
EXPIRE","USD","EURO","VARIOUS","SPIRITS","CIGARETTES","TOBACCO","
PERSONAL THINGS",
        "CHOLERA ISSUE","CHOLERA EXPIRE","BEER (CASES)","USA VISA
EXP. DATE","FATHER'S NAME","MOTHER'S NAME","USA VISA FOIL","ISSUE
COUNTRY OF S.B (FOR SANTOS CREW LIST)","USA VISA CONTROL
","COUNTRY OF RESIDENCE","CITY OF RESIDENCE","SENIORITY IN
COMPANY" };
    private  ArrayList<CrewMember> CrewData= new
CrewDataDBLists().GetCrewMembers();


    @Override
    public int getRowCount() {
        return CrewData.size();
    }

    @Override
    public int getColumnCount() {
        return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        CrewMember CrewMemData= CrewData.get(rowIndex);

        switch (columnIndex)
        {
            case 0:
                return rowIndex+1;
            case 1:
                return CrewMemData.familyName;
            case 2:
                return CrewMemData.fistName;
            case 3:
                return CrewMemData.middleName;
            case 4:
                return CrewMemData.middleInitial;
            case 5:
                return CrewMemData.rank;
            case 6:
                return CrewMemData.dateOFBirth;
            case 7:
                return CrewMemData.placeOfBirth;
            case 8:
                return CrewMemData.countryOfBirth;
            case 9:
                return CrewMemData.nationality;
```

```
case 10:
    return CrewMemData.signedOnPlace;
case 11:
    return CrewMemData.countryOfSignOn;
 case 12:
    return CrewMemData.signOnDate;
 case 13:
    return CrewMemData.lisenceNo;
case 14:
    return CrewMemData.expiryDate;
case 15:
    return CrewMemData.sex;
case 16:
    return CrewMemData.age;
case 17:
    return CrewMemData.yellowFeverIssue;
 case 18:
    return CrewMemData.yellowFeverExpire;
 case 19:
    return CrewMemData.usd;
case 20:
    return CrewMemData.euro;
case 21:
    return CrewMemData.various;
case 22:
    return CrewMemData.spirits;
case 23:
    return CrewMemData.cigarettes;
 case 24:
    return CrewMemData.tobacco;
 case 25:
    return CrewMemData.personalThings;
case 26:
    return CrewMemData.choleraIssue;
case 27:
    return CrewMemData.choleraExpire;
case 28:
    return CrewMemData.beerCases;
case 29:
    return CrewMemData.usaVisaExpDate;
case 30:
    return CrewMemData.fatherName;
case 31:
    return CrewMemData.motherName;
case 32:
    return CrewMemData.usaVisaFoil;
case 33:
    return CrewMemData.issueCountryOfSBforSanCrewL;
case 34:
    return CrewMemData.usaVisaControl;
case 35:
    return CrewMemData.countryOfResidence;
case 36:
    return CrewMemData.cityOfResidence;
case 37:
    return CrewMemData.seniorityInCompany;
```

```
            default:
                throw new UnsupportedOperationException(" not
suport");

            }

    }
    @Override
        public String getColumnName(int col){
            return columnNames[col];
        }

}
```

Πακέτο CrewDataDBcons κλάση SeamansBookJ tableModel

```
package CrewDataDBcons;
import java.util.ArrayList;
import javaapplication5.*;
import javax.swing.JTable;
import javax.swing.table.AbstractTableModel;

/**"FAMILY NAME", "FIRST NAME" ,"DATE OF BIRTH","PLACE OF
BIRTH","FATHER'S NAME","MOTHER'S NAME","No","ISSUED
PLACE","Expiry DATE"
 *
 * @author giann_000
 */
public class SeamansBookJtableModel extends AbstractTableModel {

     private  String [] columnNames= {"No","FAMILY NAME", "FIRST
NAME" ,"DATE OF BIRTH","PLACE OF BIRTH","FATHER'S NAME","MOTHER'S
NAME","No","ISSUED PLACE","Expiry DATE" };
```

```
    private  ArrayList<SeamansBook> SeamansBookData= new
CrewDataDBLists().GetSeamansBook();




    @Override
    public int getRowCount() {
        return SeamansBookData.size();
    }

    @Override
    public int getColumnCount() {
       return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
       SeamansBook seaman = SeamansBookData.get(rowIndex);


        switch (columnIndex)
        {
            case 0:
                return rowIndex+1;
            case 1:
                return seaman.CrewInfoSea.familyName;
            case 2:
                return seaman.CrewInfoSea.fistName;
            case 3:
                return seaman.CrewInfoSea.dateOFBirth;
            case 4:
                return seaman.CrewInfoSea.placeOfBirth;
            case 5:
                return seaman.CrewInfoSea.fatherName;
            case 6:
                return seaman.CrewInfoSea.motherName;
            case 7:
                return seaman.NoSea;
            case 8:
                return seaman.issuebPlaceSea;
              case 9:
             return seaman.expiryDateSea;

                 default:
                throw new UnsupportedOperationException(" not
suport");
          }


    }
    @Override
        public String getColumnName(int col){
            return columnNames[col];
        }
```

```
}
```

Πακέτο CrewDataDBcons κλάση PassportJtableModel

```
package CrewDataDBcons;

import java.util.ArrayList;
import javaapplication5.*;
import javax.swing.table.AbstractTableModel;

/**
 *
 * @author giann_000
 */
public class PassportJtableModel extends AbstractTableModel {



     private  String [] columnNames= {"No","FAMILY NAME", "FIRST
NAME" ,"DATE OF BIRTH","NATIONALITY","SEX","No","ISSUED
PLACE","Expiry DATE"};
    private  ArrayList<Passport> passData= new
CrewDataDBLists().GetPassport();



    @Override
    public int getRowCount() {
        return passData.size();
    }

    @Override
    public int getColumnCount() {
       return columnNames.length;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
       Passport passport = passData.get(rowIndex);


        switch (columnIndex)
        {
            case 0:
                return rowIndex+1;
            case 1:
                return passport.CrewInfoPass.familyName;
            case 2:
                return passport.CrewInfoPass.fistName;
            case 3:
                return passport.CrewInfoPass.dateOFBirth;
```

```
            case 4:
                return passport.CrewInfoPass.nationality;
            case 5:
                return passport.CrewInfoPass.sex;
            case 6:
                return passport.getNo();
            case 7:
                return passport.issuedPlacePass;
            case 8:
                return passport.expiryDatePass;

                 default:
                throw new UnsupportedOperationException(" not
suport");
          }


    }
    @Override
        public String getColumnName(int col){
            return columnNames[col];
        }
}
```

Πακέτο javaapplication κλάση PortDocuments

```
package javaapplication5;

import CrewDataDBcons.*;

import java.util.ArrayList;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.ParseException;
import java.util.logging.Level;
import java.util.logging.Logger;
import static org.apache.poi.hssf.usermodel.HeaderFooter.file;
import org.apache.poi.hssf.util.CellRangeAddress;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Row;
import java.util.Date;
import java.text.SimpleDateFormat;
import java.util.Locale;
import javax.swing.JFileChooser;

import org.apache.poi.ss.util.WorkbookUtil;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFSheet;
```

```
/**
 *
 * @author giann_000
 */
public class PortDocuments extends javax.swing.JFrame {

    /**
     * Creates new form PortDocuments
     */
    public PortDocuments() {
        try {
            initComponents();
            SimpleDateFormat format = new
SimpleDateFormat("dd/MM/yyyy");
            Date date = new Date();

            String dateArrival = "13/04/1989";

            date = format.parse(dateArrival);

            DateOfArrivaljDateChooser.setDate(date);

        } catch (ParseException ex) {

Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE,
null, ex);
        }

        ArrayList<CrewMember> crewMemList = new ArrayList();

        db_connection dbcon = new db_connection();
        dbcon.connect();

        crewMemList = dbcon.getCrewMembers();

        dbcon.closeConnection();

       System.out.println("Sizeeeeeeee"+crewMemList.size());
        if (crewMemList.size() >= 7) {

            jRadioButton2.setVisible(true);

        } else {
            jRadioButton2.setVisible(false);
        }
        if (crewMemList.size() >= 14 || crewMemList.size() >= 8 )
{

            jRadioButton3.setVisible(true);

        } else {
            jRadioButton3.setVisible(false);
        }
        if (crewMemList.size() >= 21 || crewMemList.size() >=15 )
{
            jRadioButton4.setVisible(true);
```

```
        } else {
            jRadioButton4.setVisible(false);
        }
        if (crewMemList.size() >= 28 || crewMemList.size() >= 22)
{

            jRadioButton5.setVisible(true);

        } else {
            jRadioButton5.setVisible(false);
        }

        if (crewMemList.size() >= 35) {

            jRadioButton6.setVisible(true);

        } else {
            jRadioButton6.setVisible(false);
        }


          jPanel2.setVisible(false);

    }

    /**
     * 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">//GEN-BEGIN:initComponents
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        documentButton = new javax.swing.JToggleButton();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        portArrival = new javax.swing.JTextField();
        portArrivedFrom = new javax.swing.JTextField();
        portOfDestination = new javax.swing.JTextField();
        DateOfArrivaljDateChooser = new
com.toedter.calendar.JDateChooser();
        jPanel1 = new javax.swing.JPanel();
        crewListRadioButton = new javax.swing.JRadioButton();
        jRadioButton8 = new javax.swing.JRadioButton();
        jRadioButton5 = new javax.swing.JRadioButton();
        jRadioButton4 = new javax.swing.JRadioButton();
        jRadioButton3 = new javax.swing.JRadioButton();
        jRadioButton2 = new javax.swing.JRadioButton();
        jRadioButton7 = new javax.swing.JRadioButton();
        jRadioButton6 = new javax.swing.JRadioButton();
```

```
        jPanel2 = new javax.swing.JPanel();
        portOfDestination1 = new javax.swing.JTextField();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        portOfDestination2 = new javax.swing.JTextField();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOS
E);

        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 18)); //
NOI18N
        jLabel1.setForeground(new java.awt.Color(0, 0, 204));
        jLabel1.setText("SELECT PORT DOCUMENTS");

        documentButton.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        documentButton.setForeground(new java.awt.Color(0, 0,
204));
        documentButton.setText("Create Documents");
        documentButton.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                documentButtonActionPerformed(evt);
            }
        });

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel2.setForeground(new java.awt.Color(0, 0, 204));
        jLabel2.setText("Port Arrival");

        jLabel3.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel3.setForeground(new java.awt.Color(0, 0, 204));
        jLabel3.setText("Port of destination");

        jLabel4.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel4.setForeground(new java.awt.Color(0, 0, 204));
        jLabel4.setText("Port Arrived from");

        jLabel5.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel5.setForeground(new java.awt.Color(0, 0, 204));
        jLabel5.setText("Date of arrival");

        DateOfArrivaljDateChooser.setDateFormatString(" d / MM /
yyyy");

        crewListRadioButton.setFont(new java.awt.Font("Tahoma",
1, 14)); // NOI18N
        crewListRadioButton.setForeground(new java.awt.Color(0,
0, 204));
        crewListRadioButton.setText("Crew List");
```

```
        crewListRadioButton.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                crewListRadioButtonActionPerformed(evt);
            }
        });

        jRadioButton8.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton8.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton8.setText("NIL List");
        jRadioButton8.addMouseListener(new
java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent
evt) {
                jRadioButton8MouseClicked(evt);
            }
            public void mouseExited(java.awt.event.MouseEvent
evt) {
                jRadioButton8MouseExited(evt);
            }
            public void mousePressed(java.awt.event.MouseEvent
evt) {
                jRadioButton8MousePressed(evt);
            }
            public void mouseReleased(java.awt.event.MouseEvent
evt) {
                jRadioButton8MouseReleased(evt);
            }
        });
        jRadioButton8.addContainerListener(new
java.awt.event.ContainerAdapter() {
            public void
componentAdded(java.awt.event.ContainerEvent evt) {
                jRadioButton8ComponentAdded(evt);
            }
            public void
componentRemoved(java.awt.event.ContainerEvent evt) {
                jRadioButton8ComponentRemoved(evt);
            }
        });
        jRadioButton8.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton8ActionPerformed(evt);
            }
        });
        jRadioButton8.addAncestorListener(new
javax.swing.event.AncestorListener() {
            public void
ancestorMoved(javax.swing.event.AncestorEvent evt) {
            }
            public void
ancestorAdded(javax.swing.event.AncestorEvent evt) {
```

```
            }
            public void
ancestorRemoved(javax.swing.event.AncestorEvent evt) {
                jRadioButton8AncestorRemoved(evt);
            }
        });
        jRadioButton8.addKeyListener(new
java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jRadioButton8KeyPressed(evt);
            }
            public void keyReleased(java.awt.event.KeyEvent evt)
{
                jRadioButton8KeyReleased(evt);
            }
        });

        jRadioButton5.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton5.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton5.setText("Crew Effects P.4");
        jRadioButton5.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton5ActionPerformed(evt);
            }
        });

        jRadioButton4.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton4.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton4.setText("Crew Effects P.3");
        jRadioButton4.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton4ActionPerformed(evt);
            }
        });

        jRadioButton3.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton3.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton3.setText("Crew Effects P.2");
        jRadioButton3.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton3ActionPerformed(evt);
            }
        });
```

```
        jRadioButton2.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton2.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton2.setText("Crew Effects P.1");
        jRadioButton2.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton2ActionPerformed(evt);
            }
        });

        jRadioButton7.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton7.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton7.setText("Vaccination List");
        jRadioButton7.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton7ActionPerformed(evt);
            }
        });

        jRadioButton6.setFont(new java.awt.Font("Tahoma", 1,
14)); // NOI18N
        jRadioButton6.setForeground(new java.awt.Color(0, 0,
204));
        jRadioButton6.setText("Crew Effects P.5");
        jRadioButton6.addActionListener(new
java.awt.event.ActionListener() {
            public void
actionPerformed(java.awt.event.ActionEvent evt) {
                jRadioButton6ActionPerformed(evt);
            }
        });

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

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel1Layout.createSequentialGroup()

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING, false)
                    .addComponent(jRadioButton6,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                    .addComponent(crewListRadioButton)
                    .addComponent(jRadioButton2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jRadioButton3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jRadioButton4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jRadioButton5,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jRadioButton7)
                    .addComponent(jRadioButton8))
                .addGap(25, 25, 25))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(crewListRadioButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)
                .addComponent(jRadioButton2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)
                .addComponent(jRadioButton3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)
                .addComponent(jRadioButton4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)
                .addComponent(jRadioButton5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)
                .addComponent(jRadioButton6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)
                .addComponent(jRadioButton7)
                .addGap(8, 8, 8)
                .addComponent(jRadioButton8)
                .addContainerGap(46, Short.MAX_VALUE))
        );

        jLabel7.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel7.setForeground(new java.awt.Color(0, 0, 204));
        jLabel7.setText("Passengers");
```

```
        jLabel8.setFont(new java.awt.Font("Tahoma", 1, 14)); //
NOI18N
        jLabel8.setForeground(new java.awt.Color(0, 0, 204));
        jLabel8.setText("Pets / Plans");

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                .addContainerGap(13, Short.MAX_VALUE)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING, false)
                    .addComponent(jLabel8,
javax.swing.GroupLayout.DEFAULT_SIZE, 133, Short.MAX_VALUE)
                    .addComponent(jLabel7,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.LEADING)
                    .addComponent(portOfDestination1,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(portOfDestination2,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(102, 102, 102))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
                    .addComponent(jLabel7,
javax.swing.GroupLayout.PREFERRED_SIZE, 27,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(portOfDestination1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.BASELINE)
```

```
                    .addComponent(jLabel8,
javax.swing.GroupLayout.PREFERRED_SIZE, 25,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(portOfDestination2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(0, 209, Short.MAX_VALUE))
        );

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

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(48, 48, 48)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING, false)
                            .addComponent(jLabel4,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jLabel5,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 129,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING, false)
                            .addComponent(portArrival)
                            .addComponent(portArrivedFrom,
javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(DateOfArrivaljDateChooser,
javax.swing.GroupLayout.DEFAULT_SIZE, 155, Short.MAX_VALUE)
                            .addComponent(portOfDestination))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED, 119, Short.MAX_VALUE))

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

```
.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING, false)
                    .addComponent(documentButton,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGap(63, 63, 63))
            .addGroup(layout.createSequentialGroup()
                .addGap(236, 236, 236)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 246,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(100, 100, 100)
                        .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(27, 27, 27)
                        .addComponent(documentButton,
javax.swing.GroupLayout.PREFERRED_SIZE, 36,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(27, 27, 27)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 48,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(45, 45, 45)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                            .addComponent(jLabel2)
```

```
                            .addComponent(portArrival,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELAT
ED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                            .addComponent(jLabel4)
                            .addComponent(portArrivedFrom,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.LEADING)
                            .addComponent(jLabel5)

.addComponent(DateOfArrivaljDateChooser,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNREL
ATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alig
nment.BASELINE)
                            .addComponent(jLabel3)
                            .addComponent(portOfDestination,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(18, 18, 18)
                        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(55, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void
crewListRadioButtonActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_crewListRadioButtonActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_crewListRadioButtonActionPerformed
```

```
    private void
jRadioButton2ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton2ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton2ActionPerformed

    private void
jRadioButton3ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton3ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton3ActionPerformed

    private void
jRadioButton4ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton4ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton4ActionPerformed

    private void
jRadioButton5ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton5ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton5ActionPerformed

    private void
jRadioButton6ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton6ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton6ActionPerformed

    private void
jRadioButton7ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton7ActionPerformed
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton7ActionPerformed

    private void
jRadioButton8ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jRadioButton8ActionPerformed

        if(jRadioButton8.isSelected()){
             jPanel2.setVisible(true);

        }else{
              jPanel2.setVisible(false);
        }


    }//GEN-LAST:event_jRadioButton8ActionPerformed

    private void
documentButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_documentButtonActionPerformed

        String PortArrival;
        String PortArrivedFrom;
        String DateOfArrival;
```

```
        String PortOfDestination;

        SimpleDateFormat formatoBrasileiro = new
SimpleDateFormat("dd/MM/yyyy");
        Date date = new Date();

        //String dateArrival="13/04/1989";
        ArrayList<CrewMember> crewMem = new ArrayList();
        ArrayList<Passport> passportList = new ArrayList();
        ArrayList<Vessel> vesselList = new ArrayList();
        HSSFWorkbook workbook = null;

        CrewDataDBLists CrewDataDBDb = new CrewDataDBLists();
        db_connection condb = new db_connection();
        condb.connect();

        crewMem = CrewDataDBDb.GetCrewMembers();
        passportList = CrewDataDBDb.GetPassport();
        vesselList = condb.getVesselDB();

        condb.closeConnection();
        boolean success = (new
File("C:/JavaApplication5/port")).mkdirs();

        try {

            workbook = new HSSFWorkbook(new
FileInputStream("C:\\\\JavaApplication5\\\\Excels\\\\CrewList2.xl
s"));

            Sheet sheet = workbook.getSheet("crew");

            for (int i = 0; i < crewMem.size(); i++) {
                Cell cellCount = sheet.getRow(i + 10).getCell(0);
                cellCount.setCellValue(i + 1);

                Row row = sheet.getRow(10 + i);
                Cell cell = row.getCell(1);

cell.setCellValue(crewMem.get(i).getFamilyName().toString());

                Cell cellFirtName = sheet.getRow(i +
10).getCell(2);

cellFirtName.setCellValue(crewMem.get(i).getFistName().toString()
);

                Cell cellMiddleName = sheet.getRow(i +
10).getCell(3);

cellMiddleName.setCellValue(crewMem.get(i).getMiddleName().toStri
ng());

                Cell cellSex = sheet.getRow(i + 10).getCell(4);
                cellSex.setCellValue(crewMem.get(i).getSex());

                Cell cellRank = sheet.getRow(i + 10).getCell(5);
```

```
                cellRank.setCellValue(crewMem.get(i).getRank());

                Cell cellNationality = sheet.getRow(i +
10).getCell(6);

cellNationality.setCellValue(crewMem.get(i).getNationality());

                Cell cellBirth = sheet.getRow(i + 10).getCell(7);

cellBirth.setCellValue(crewMem.get(i).getDateOFBirth());

                Cell cellPlaceOfBirth = sheet.getRow(i +
10).getCell(8);

cellPlaceOfBirth.setCellValue(crewMem.get(i).getPlaceOfBirth());

                Cell cellPassportNo = sheet.getRow(i +
10).getCell(9);

cellPassportNo.setCellValue(passportList.get(i).getNo());

                Cell cellExpiryDate = sheet.getRow(i +
10).getCell(10);

cellExpiryDate.setCellValue(crewMem.get(i).getExpiryDate());

            }

        } catch (FileNotFoundException ex) {

Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (IOException ex) {

Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE,
null, ex);
        }

        try {
            // boolean success = (new
File("C:/Users/giann_000/Documents/NetBeansProjects/JavaApplicati
on5/port")).mkdirs();
            File file = new
File("C:\\JavaApplication5\\port\\Document1.xls");

            FileOutputStream output = new FileOutputStream(file);
            workbook.write(output);

            output.close();

        } catch (Exception e) {

        }
        //   MOD2   DIV4

        int crewnumber = 30;
        int crewnumber1 = 30;
```

```
        crewnumber = crewnumber % 7;
        crewnumber1 = crewnumber1 / 7;
        System.out.println("MOD" + crewnumber + "   DIV" +
crewnumber1);

        int crewEffectsCounter = 0;

        if (crewMem.size() % 7 == 0) {
            crewEffectsCounter = crewMem.size() / 7;
        } else {
            crewEffectsCounter = (crewMem.size() / 7) + 1;
        }

        int counterMem = 0;
        int count = 0;

        for (int i = 0; i < crewEffectsCounter; i++) {

            if (crewEffectsCounter - i == 1) {
                if (crewMem.size() % 7 == 0) {
                    counterMem = crewnumber % 7;
                } else {
                    counterMem = 7;
                }

            } else {
                counterMem = 7;
            }

            try {

                workbook = new HSSFWorkbook(new
FileInputStream("C:\\\\Users\\\\giann_000\\\\Documents\\\\NetBean
sProjects\\\\JavaApplication5\\\\Excels\\\\crewEffect.xls"));

                Sheet sheet = workbook.getSheet("crewEffect");

                int y = 0;

                if (i > 1) {
                    count = count + 7;

                }

                for (int k = 0; k < counterMem; k++) {

                    Cell cellCount = sheet.getRow(y +
15).getCell(0);
                    cellCount.setCellValue(k + 1 + count);

                    Row row = sheet.getRow(15 + y);
                    Cell cell = row.getCell(1);
                    cell.setCellValue(crewMem.get(k +
count).getFamilyName().toString());

                    Cell cellFirtName = sheet.getRow(y +
16).getCell(1);
```

```
                    cellFirtName.setCellValue(crewMem.get(k +
count).getFistName().toString());

                    Cell cellrank = sheet.getRow(y +
15).getCell(2);
                    cellrank.setCellValue(crewMem.get(k +
count).getRank());

                    Cell cellSpirits = sheet.getRow(y +
15).getCell(3);
                    cellSpirits.setCellValue(crewMem.get(k +
count).getSpirits());

                    Cell cellCigar = sheet.getRow(y +
15).getCell(4);
                    cellCigar.setCellValue(crewMem.get(k +
count).getCigarettes());

                    Cell celltobacco = sheet.getRow(y +
15).getCell(5);
                    celltobacco.setCellValue(crewMem.get(k +
count).getTobacco());

                    Cell cellPersonalTh = sheet.getRow(y +
15).getCell(6);
                    cellPersonalTh.setCellValue(crewMem.get(k +
count).getPersonalThings());

                    y = y + 5;

                }

            } catch (FileNotFoundException ex) {

Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE,
null, ex);
            } catch (IOException ex) {

Logger.getLogger(PortDocuments.class.getName()).log(Level.SEVERE,
null, ex);
            }

            try {
                // boolean success = (new
File("C:/Users/giann_000/Documents/NetBeansProjects/JavaApplicati
on5/port")).mkdirs();
                File file = new
File("C:\\Users\\giann_000\\Documents\\NetBeansProjects\\JavaAppl
ication5\\port\\CrewEffecs" + i + 1 + ".xls");

                FileOutputStream output = new
FileOutputStream(file);
                workbook.write(output);

                output.close();

            } catch (Exception e) {
```

```
            }

        }

        Date dateFromDateChooser =
DateOfArrivaljDateChooser.getDate();
        String dateString = String.format("%1$td-%1$tm-%1$tY",
dateFromDateChooser);
        System.out.println("DATE  : " + dateString);


    }//GEN-LAST:event_documentButtonActionPerformed

    private void jRadioButton8KeyPressed(java.awt.event.KeyEvent
evt) {//GEN-FIRST:event_jRadioButton8KeyPressed

    }//GEN-LAST:event_jRadioButton8KeyPressed

    private void jRadioButton8KeyReleased(java.awt.event.KeyEvent
evt) {//GEN-FIRST:event_jRadioButton8KeyReleased




    }//GEN-LAST:event_jRadioButton8KeyReleased

    private void
jRadioButton8MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jRadioButton8MouseClicked
        // TODO add your handling code here:
    }//GEN-LAST:event_jRadioButton8MouseClicked
  int m=0;
    private void
jRadioButton8MousePressed(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jRadioButton8MousePressed
    //


    }//GEN-LAST:event_jRadioButton8MousePressed

    private void
jRadioButton8MouseReleased(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jRadioButton8MouseReleased

    }//GEN-LAST:event_jRadioButton8MouseReleased

    private void
jRadioButton8MouseExited(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jRadioButton8MouseExited

    }//GEN-LAST:event_jRadioButton8MouseExited

    private void
jRadioButton8AncestorRemoved(javax.swing.event.AncestorEvent evt)
{//GEN-FIRST:event_jRadioButton8AncestorRemoved
```

```
    }//GEN-LAST:event_jRadioButton8AncestorRemoved

    private void
jRadioButton8ComponentRemoved(java.awt.event.ContainerEvent evt)
{//GEN-FIRST:event_jRadioButton8ComponentRemoved

    }//GEN-LAST:event_jRadioButton8ComponentRemoved

    private void
jRadioButton8ComponentAdded(java.awt.event.ContainerEvent evt)
{//GEN-FIRST:event_jRadioButton8ComponentAdded

    }//GEN-LAST:event_jRadioButton8ComponentAdded

    /**
     * @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/pl
af.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(PortDocuments.class.getName())
.log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(PortDocuments.class.getName())
.log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(PortDocuments.class.getName())
.log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex)
{

java.util.logging.Logger.getLogger(PortDocuments.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 PortDocuments().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private com.toedter.calendar.JDateChooser
DateOfArrivaljDateChooser;
```