ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΘΕΜΑ : «ΑΕΡΟΠΟΡΙΚΗ ΕΤΑΙΡΕΙΑ»

Φοιτητες: *Γερακιανάκης Νικόλαος, 3949*

Παντελεήμων Μαρινάκης, 3781

Επιβλέπων*: Παπαδάκης Νικόλαος*

Ηράκλειο Κρήτης, 2019

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Table of Figures

ΠΕΡΙΛΗΨΗ

Αρχικά κάνουμε μια μικρή ιστορική αναδρομή όσο αφορά το αεροπλάνο και στη συνέχεια περιγράφουμε επιγραμματικά τα βασικά σημεια αυτού. Αποδεικνύουμε γιατί το αεροπλάνο ειναι απαραίτητο στην εποχή μας ενώ στη συνέχεια προχωράμε στο να επεξηγήσουμε το πως θα εγκαταστήσουμε τα βασικά προγραμματα που θα μας βοηθήσουν να επιλύσουμε την εφαρμογή που πρόκειται να υλοποιήσουμε όπως μας ζητήθηκε. Θα χρησιμοποιηθεί η γλώσσα προγραματισμού Java καθως και οι βάσεις δεδομένων MySQL. Στη συνέχεια θα περιγράψουμε βήμα προς βήμα το πώς υλοποιήσαμε την εφαρμογή.

ΕΥΧΑΡΙΣΤΙΕΣ

- Θα θέλαμε να ευχαριστήσουμε τον κ. Παπαδάκη Νικόλαο επιβλέποντα καθηγητή της πτυχιακής μας εργασίας για την άψογη συνεργασία που είχαμε.
- Ευχαριστούμε όλους τους καθηγητές και τον κάθε έναν τους ξεχωριστά για τη θετική διάθεση να μας λύσουν όλα τα ερωτήματα που μας γεννούσε η σχολή σε επίπεδο γνώσεων.
- Τέλος, «αλλά όχι τελευταίοι - λέει πολύ σωστά μια παροιμία», ευχαριστούμε τις οικογένειές μας για όλη τη στήριξη που μας παρείχαν καθ' όλη τη διάρκεια σπουδών σε όλους τους τομείς και που δέχτηκαν να πάρουν όλη αυτή την πίεση πάνω τους με σκοπό να έχουμε ένα καλύτερο μέλλον.

ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Ως αεροπλάνο θεωρείται ενα μηχανοκίνητο αεροσκάφος που διαθέτει σταθερές πτέρυγες το οποίο χρησιμοποιεί για την ώθηση του έναν έλικα ή έναν κινητήρα τύπου αεριώθησης. Φυσικά υπάρχουν πάρα πολλά είδη αεροσκαφών με διάφορα σχέδια και με διαφορετικές διατάξεις πτερύγων(Gibbs, Charles, 1959).

Τα αεροπλάνα θεωρούνται πολύ χρήσιμα στις ημέρες μας καθώς μπορούν να προσφέρουν καταρχάς μεταφορά επιβατών αλλά και αλλων πραγμάτων πχ εμπορικά είδη όπως αγαθά και επιπρόσθετα είναι χρήσιμα στον στρατό και για ταξίδια αναψυχής. Επομένως η εμπορευματική αεροπορία θεωρείται (Gibbs, Charles, 1959) από μόνη της μια τεράστια βιομηχανία και τα περισσότερα από αυτά (αεροπλάνα) καθοδηγούνται από έναν κυβερνήτη. Υπάρχουν φυσικά αρκετά είδη αεροσκαφών που κυβερνούνται απο από υπολογιστή εξ αποστάσεως (Benedetti, 2003).

Το πρώτο αεροπλάνο κατασκευάστηκε απο τους αδελφούς Ραίτ που πέταξαν το πρώτο αεροπλάνο το 1903 και το κατασκεύασαν σύμφωνα με τα σχέδια του Τζώρτζ Κέιλι που χρονολογούνται απο το 1779. Στη συνέχεια ανάμεσα στα έτη 1867 και 1896(Encyclopaedia Britannica) ο Γερμανός Λίλιενταλ μέτρησε τις βαρύτερες απο τον αέρα πτήσεις και παρόλες τις δυσκολίες. Να τονίσουμε ότι τα αεροσκάφη συνέχισαν να εξελίσσονται αλλα και να δέχονται κατάλληλη βελτίωση κατά την διάρκεια του Πρώτου Παγκοσμίου Πολέμου (Encyclopaedia Britannica)

Το πρώτο γνωστό αεροπλάνο επιβατικού τύπου που χρησιμοποίησε κινητήρα τύπου αεριώθησης ονομάστηκε de Havilland Comet και ξεκίνησε τις πτήσεις του το 1905 ενώ το γνωστό Boeing 747 θεωρήθηκε ως το πιο γνωστό αλλά και επιτυχημένο αεροσκάφος που διέθετε κινητήρα τύπου αεριώθησης, εμπορικού τύπου και χρησιμοποιήθηκε τα έτη 1958 μέχρι και το 2013(Beaumont, 1942)

Πηγαίνοντας λίγο πίσω στην χρονολογία του 400 π.Χ παρατηρούμε οτι στην Αρχαία Ελλάδα ο γνωστός Αρχύτας κατασκεύασε και σχεδίασε την ώς πλέον ευρύτερα γνωστή, ιπτάμενη συσκευή που ήταν και αυτοκινούμενη. Διέθετε σχήμα πουλιού ενώ για την ώθησή του χρησιμοποιούσε μηχανή και πέταξε συνολικό μήκος στα 200 μέτρα (tmth, 2008).

Από την άλλη πλευρά ο γνωστός ζωγράφος και πανεπιστήμονας Λεονάρντο ντα Βίντσι στην διάρκεια της ζωής του ερεύνησε ένα αεροσκάφος που υποτίθεται ηταν ανθρωποκίνητο στο γνωστό έργο του Κώδιξ στο Πέταγμα των Πουλιών (White, 1961).

Πιο μετά στο ετος 1799 ο Τζώρτζ Κέιλι αρχικά όρισε την βασική έννοια του σύγχρονου αεροσκάφους ως μια μηχανή που διέθετε πτερύγια και ηταν ιπτάμενη. Τα πτερύγια θεωρούνται συστήματα ελέγχου, προώθησης και άνωσης (Dwyer).

Από την άλλη πλευρά, ο γνωστός Χάιραμ Μάξιμ δημιούργησε ένα αεροσκάφος διέθετε βάρος 3.5 τόνους ενώ το μήκος των πτερύγων του κατα την διάρκεια του ανοίγματός τους ήταν στα 34 μέτρα. Για τον τρόπο λειτουργίας τους χρησιμοποιήθηκαν δύο ατμομηχανές 360 ίππων που χρησιμοποιούνταν για να κινούν δύο έλικες (The Journal of San Diego History, 1968).

Έτσι, το 1894, η μηχανή του δοκιμάστηκε με εναέριες ράγες για να ελεγχθεί η άνοδος του. Πιο αναλυτικά, η δοκιμή έδειξε πως είχε πολύ βάρος για να ανυψωθεί. Το σκάφος ήταν ανεξέλεγκτο (Becker, 1967).

Πιο μετά, συγκεκριμένα κατά τη δεκαετία του 1890 ο Χαργκρειβ ερεύνησε την δομή των φτερών (Inglis, 1983) των αεροσκαφών και δημιούργησε ένα «χαρταετό» που θεωρητικά θα μπορούσε να σηκώσει με άνεση το ανθρώπινο βάρος. Τα σχέδια του

Χαργκρειβ υιοθετήθηκαν αλλα και υλοποιήθκκαν απο πολλές εταιρείες ανα τον κόσμο (Becker, 1967).

ΟΙ ΠΡΩΤΕΣ ΠΤΗΣΕΙΣ

Ήδη απο το 1905 το γνωστό αεροσκάφος Wright Flyer III, διέθετε την ικανότητα του πλήρους ελέγχου στον αέρα αλλα και μια σταθερότητα πτήσης για εναν αρκετό χρόνο ενώ βρίσκονταν σε πτήση. Η διαπίστωση των αδελφών Ραιτ ήταν τελικά να συνεχίσουν τις επανδρωμένες πτήσεις (Benedetti, 2003).

Επιπρόσθετα το έτος 1906 ο Σάντους-Ντουμό τελικά πραγματοποίησε την γνωστή πρώτη πτήση αεροπλάνου δίχως να χρειαστεί την βοήθεια του καταπέλτη θέτοντας έτσι ενα παγκόσμιο, πρώτο ρεκόρ που τράβηξε τα βλέμματα της Αερολέσχης της Γαλλίας μιας και πέταξε κοντά στα 220 μέτρα σε χρόνο λιγότερο απο 22 δευτερόλεπτα (Crouch, 1982).

Επομένως, το 1908 ήρθε ως πρώτο σχέδιο πλήρους αεροσκάφους με σύγχρονη διάταξη ελκυστήρων (μονοπλάνων) το αεροσκάφος Bleriot VIII. Το αεροσκάφος αυτό διέθετε (Beaumont, 1942) σταθερές κινούμενες επιφάνειες στην ουρά του. Έτσι ο έλεγχος της απόκλισης και το βήμα γινόταν πιο πιο εύκολα, διαμέσου μιας κατάλληλης στρέβλωσης των πτερύγων, με την βοήθεια του πιλότου(Crouch, 1982).

Επιπρόσθετα, ήδη στην Ρουμανία το 1910 δημιουργήθηκε το αεροσκάφος A.Vlaicunr. I (century-of-flight, 2016).

Το αεροσκάφος αυτό κατασκευάστηκε από έναν δοκό αλουμινίου μήκους δέκα μέτρων πάνω στον οποίο στηρίζοταν ολόκληρο το αεροσκάφος, ενώ στην κυκλοφορία διατέθηκαν δέκα αεροσκάφη για την Ρουμανική Πολεμική Αεροπορία.

Είναι γνωστό ότι ο πρώτος παγκόσμιος πόλεμος θεωρήθηκε ο βασικός χώρος κατάλληλων δοκιμών όσο αφορά την χρησιμότητα του αεροπλάνου πολεμική μηχανή. Επομένως στη συνέχεια αναδείχθηκαν σε μηχανές πολέμου καθώς μπορούσαν να προκαλέσουν αρκετή ζημιά στον εχθρό (Spaight, 1914).

Μετά τον πρώτο παγκόσμιο πόλεμο η τεχνολογία των αεροπλάνων συνέχισε να υποδέχεται βελτίωση αλλά και εξέλιξη. Επιπρόσθετα, οι Μπράουν και Αλκοκ με την βοήθεια αεροσκάφους μπόρεσαν να διασχίσουν τον Ατλαντικό ωκεανό το 1919 χωρίς να κάνουν στάσεις πουθενά (Paur, 2010).

Να προσθέσουμε οτι τα αεροπλάνα συμμετείχαν σχεδόν σε όλες τις μεγάλες μάχες του Β΄Παγκοσμίου Πολέμου (Paur, 2010).

ΤΡΟΠΟΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΩΝ ΑΕΡΟΠΛΑΝΩΝ

ΤΑ ΒΑΣΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Τα βασικά μέρη που δομούν ενα αεροσκάφους ονομάζεται κύριο σώμα. Τα τμήματα αυτά μπορούν και να έχουν διαφορές ανάλογα με τον τύπο του αεροσκάφους αλλά και με τον βασικό σκοπό που επιτελεί (Federal Aviation Administration).

Μιας και οι ταχύτητες των αεροσκαφών αυξανόταν συνεχώς, τα τμήματα των αεροπλάνων σε βάθος χρόνου έγιναν όλα μεταλλικά. Στην εποχή μας υπάρχουν αεροπλάνα που ειναι κατασκευασμένα απο μέταλλο αλλά και συνθετικά υλικά (Federal Aviation Administration).

Στα βασικά τμήματα του αεροσκάφους περιλαμβάνονται τα παρακάτω(NASA):

- Μια ή και αρκετές περισσότερες οριζόντιες πτέρυγες

    - Η πτέρυγα έχει την ικανότητα να εκτρέπει τον αέρα προς τα κάτω μιας και το αεροσκάφος κινείται μπροστά.
    - Ο βασικός σκοπός της πτέρυγας είναι να παρέχει κατάλληλη σταθερότητα με σκοπό να μπορεί να αποφευχθούν οι κινήσεις προς τα δεξία ή αριστερά ενόσω το αεροσκάφος βρίσκεται σε πτήση.
- Μια άτρακτος που διαθέτει ένα αρκετά λεπτό και μακρύ σώμα με έντονες στρογγυλλεμένες και κωνικές άκρες με σκοπό την παροχή αεροδυναμικής ικανότητας στα τμήματα του αεροσκάφους.
- Ένας κάθετος σταθεροποιητής που ουσιαστικά είναι μια κάθετη επιφάνεια που ομοιάζει με φτερό και ειναι κατάλληλα τοποθετημένη στο πίσω μέρος του αεροπλάνου ενώ πολλές φορές μπορεί να προεξέχει απο αυτό.
- Ένας οριζόντιος σταθεροποιητής που τοποθετείται στην ουρά κοντά στον κάθετο σταθεροποιητή
- Τροχοί προσγείωσης

Η ΔΟΜΗ ΤΩΝ ΠΤΕΡΥΓΩΝ

Τα αεροσκάφη συνήθως περιέχουν επιφάνειες πτερύγων που ειναι ευέλικτες αλλά και εκτείνονται προς ένα πλαίσιο που στη συνέχεια γίνονται άκαμπτες λόγω των δυνάμεων ανώσεων που ασκούνται απο τον άερα ακριβώς επάνω τους (NASA).

Επομένως, τα βασικά δομικά στοιχεία ειναι οι δοκοί που μπορούν να είναι ένας ή και περισσότεροι, εκτείνονται απο άκρη σε άκρη και διαθέτουν πολλαπλές ραβδώσεις (NASA).

Ειδικά τα παλαιότερα χρόνια οι πρώτοι κινητήρες των αεροσκαφών είχαν ελάχιστη ισχύ σε σχέση με το σήμερα που αυτό είχε σαν αποτέλεσμα, βασικά τμήματα των αεροτομών να ήταν αρκετά λεπτά με αποτέλεσμα να μην μπορούν να σηκώσουν ενα ισχυρό και βαρύ πλαίσιο στην επιφάνεια τους (NASA).

Επομένως, στα μετέπειτα χρόνια, προστέθηκαν εξωτερικοί δοκοί αντιστήριξης και καλώδια.

Η ΔΙΑΤΑΞΗ ΤΩΝ ΠΤΕΡΥΓΩΝ

Αρκετές φορές ο αριθμός αλλά και το σχήμα των πτερύγων διαφέρει κατά πολύ όσο αφορά τον τύπου του αεροσκάφους που τις διαθέτουν. Επομένως, μια πτέρυγα μπορεί να θεωρηθεί πλήρους ανοίγματος ή και όχι ανάλογα με τον τύπο της (flightglobal, 1975).

Όταν κατά τη διάρκεια του 1920 μέχρι 1930 η ισχύς του κινητήρα αυξήθηκε τότε το βασικό και ως κατά συνέπεια αστήρικτο μονόπλανο θεωρήθηκε ως ο πιο κοινός τύπους αεροπλάνου που χρησιμοποιήσουσε μηχανοκίνηση για την κίνηση του (flightglobal, 1975).

Μια πτερύγα για να μπορέσει να θεωρηθεί αεροδυναμικά σταθερή θα πρέπει να είναι καταρχάς ευθεία, με αρκετά μεγάλη έκταση ενώ να έχει μικρή χορδή. Επομένως για να είναι και αποτελεσματική πρέπει να έχει χαμηλό βάρος κατάλληλο για να μπορέσει να παρέχει την άνωση που χρειάζεται.

Όσο αφορά τις διηχητικές ταχύτητες θα πρέπει η σάρωση της μίας πτέρυγας μπροστά ή και πίσω να έχει την ικανότητα να μειώνει την αντίσταση λόγω των υπερηχητικών ωστικών κυμάτων που αναπτύσσονται κατά τη διάρκεια της πτήσης. Οπότε,

θεωρούμε ως πτερύγα σάρωσης μια ευθεία πτέρυγα που διαθέτει την ικανότητα να κινείται πίσω και μπροστά (flightglobal, 1975).

Επίσης, μπορούμε να θεωρήσουμε πως μια πτέρυγα τέτοιου είδους είναι η πτέρυγα Rogallo που δίνει τις κατάλληλες αεροδυναμικές δυνάμεις και χρησιμοποιείται όμως σε αρκετά ελαφριά σε βάρος αεροπλάνα ή και σε χαρταετούς (flightglobal, 1975).

Από την άλλη πλευρά, μια πτέρυγα μπορεί να θεωρηθεί ως υπερηχητική εάν διαθέτει χαμηλή αντίσταση σε συνδυασμό με υψηλή δύναμη, ενώ βρίσκει χρησιμότητα στα πολύ γρήγορα αεροσκάφη που διαθέτουν αεριωθούμενο κινητήρα (History-NASA).

Τέλος, ονομάζουμε μια πτέρυγα γεωμετρική όταν αλλάζει το σχήμα της κατά τη διάρκεια της πτήσεως σε άλλο σχήμα. Δηλαδή, οι μεταβλητές πτέρυγες σαρώσεως αλλάζουν κατάλληλα μέσω μιας κατάλληλης ευθείας διάταξης που υπάρχει στο αεροσκάφος κατά την διάρκεια προσγείωσης και της απογείωσης σε μια άλλη διάταξη χαμηλής αντιστάσεως, όταν πρόκειται για πτήση υψηλής ταχύτητας (History-NASA).

ΑΕΡΟΠΛΑΝΟ & ΠΕΡΙΒΑΛΛΟΝ

Όπως όλες οι δραστηριότητες που αφορούν την καύση, τα ατμοσφαιρικά καύσιμα απελευθερώνουν αιθάλη και άλλους ρύπους στην ατμόσφαιρα. Παράγονται επίσης αέρια θερμοκηπίου όπως το διοξείδιο του άνθρακα (CO_2).

Επιπλέον, υπάρχουν περιβαλλοντικές επιπτώσεις ειδικά για τα αεροπλάνα (Penner Joyce, Lister, Griggs, Dokken, McFarland, 1999)

- Τα αεροπλάνα που λειτουργούν σε μεγάλα υψόμετρα κοντά στην τροπόπαυση εκπέμπουν αερολύματα και αφήνουν τα κονιάματα, και τα δύο μπορούν να αυξήσουν τον σχηματισμό νέφους κύκλων - το κάλυμμα του νέφους μπορεί

να έχει αυξηθεί έως και 0,2% από τη γέννηση της αεροπορίας (Penner Joyce, Lister, Griggs, Dokken, McFarland, 1999)

- Τα αεροπλάνα που λειτουργούν σε μεγάλα υψόμετρα κοντά στην τροπόπαυση μπορούν επίσης να απελευθερώσουν χημικές ουσίες που αλληλεπιδρούν με τα αέρια του θερμοκηπίου σε αυτά τα ύψη, ιδιαίτερα τις ενώσεις αζώτου, τα οποία αλληλεπιδρούν με το όζον αυξάνοντας τις συγκεντρώσεις του όζοντος.
- Τα περισσότερα αεροσκάφη ελαφρού εμβόλου καίγονται avgas, το οποίο περιέχει τετρααιθυλαλεάδη (TEL). Ορισμένοι εμβολοφόροι κινητήρες χαμηλότερης συμπίεσης μπορούν να λειτουργήσουν σε αμόλυβδη κινητήρες στροβίλων και στροβίλων και πετρελαιοκινητήρες - καθένα από τους οποίους δεν απαιτούν μόλυβδο (Grewe, Brunner, Dameris, Grenfell, Hein, Shindell, Staehelin, 2001).
- Ένας άλλος περιβαλλοντικός αντίκτυπος των αεροπλάνων είναι η ηχορύπανση (Penner Joyce, Lister, Griggs, Dokken, McFarland, 1999).

Η ΕΝΝΟΙΑ ΚΑΙ Η ΧΡΗΣΙΜΟΤΗΤΑ ΤΩΝ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Μια βάση δεδομένων είναι μια οργανωμένη συλλογή δεδομένων, που αποθηκεύεται αλλά και προσπελάζεται ηλεκτρονικά από ένα σύστημα υπολογιστή.

Το σύστημα διαχείρισης βάσεων δεδομένων (DBMS) είναι το λογισμικό που αλληλεπιδρά με τους τελικούς χρήστες, τις εφαρμογές αλλά και την ίδια τη βάση δεδομένων για την καταγραφή και ανάλυση των δεδομένων.

Το λογισμικό DBMS περιλαμβάνει επιπλέον τις βασικές διευκολύνσεις που παρέχονται για τη διαχείριση της βάσης δεδομένων. Το συνολικό άθροισμα της βάσης δεδομένων, του ΣΔΒΔ και των σχετικών εφαρμογών μπορεί να αναφέρεται ως σύστημα βάσης δεδομένων.

Οι επιστήμονες υπολογιστών μπορούν να ταξινομούν τα συστήματα διαχείρισης βάσεων δεδομένων σύμφωνα με τα μοντέλα βάσης δεδομένων που υποστηρίζουν. Οι σχετικές βάσεις δεδομένων κατέστησαν κυρίαρχη στη δεκαετία του '80.

Αυτά τα δεδομένα μοντέλων ως σειρές και στήλες σε μια σειρά πινάκων και η **συντριπτική πλειοψηφία χρησιμοποιεί SQL** για τη γραφή και την αναζήτηση δεδομένων.

Στη δεκαετία του 2000, οι μη σχεσιακές βάσεις δεδομένων έγιναν δημοφιλή, που αναφέρονται ως NoSQL επειδή χρησιμοποιούν διαφορετικές γλώσσες ερωτημάτων.

ΒΑΣΙΚΕΣ ΛΕΙΤΟΥΡΓΙΕΣ ΤΩΝ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Η πρόσβαση στα δεδομένα αυτά παρέχεται συνήθως από ένα σύστημα διαχείρισης βάσεων δεδομένων (DBMS) που αποτελείται από ένα ολοκληρωμένο σύνολο λογισμικού που επιτρέπει στους χρήστες να αλληλεπιδρούν με μία ή περισσότερες βάσεις δεδομένων και παρέχει πρόσβαση σε όλα τα δεδομένα που περιέχονται στη βάση δεδομένων μπορεί να υπάρχουν που περιορίζουν την πρόσβαση σε συγκεκριμένα δεδομένα (Bachman, 1973).

Εκτός του κόσμου της επαγγελματικής τεχνολογίας των πληροφοριών, ο όρος βάση δεδομένων χρησιμοποιείται συχνά για να αναφέρεται σε οποιαδήποτε συλλογή σχετικών δεδομένων μιας και το μέγεθος και οι απαιτήσεις χρήσης συνήθως απαιτούν τη χρήση ενός συστήματος διαχείρισης βάσεων δεδομένων (Bachman, 1973).

Τα υπάρχοντα ΣΔΒΔ παρέχουν διάφορες λειτουργίες που επιτρέπουν τη διαχείριση μιας βάσης δεδομένων και των δεδομένων της, τα οποία μπορούν να ταξινομηθούν σε τέσσερις κύριες λειτουργικές ομάδες (Bachman, 1973):

- Ορισμός δεδομένων - Δημιουργία, τροποποίηση αλλά και κατάργηση ορισμών που καθορίζουν την οργάνωση των δεδομένων.
- Ενημέρωση - Εισαγωγή, τροποποίηση και διαγραφή των πραγματικών δεδομένων (Beynon-Davies, 2003).
- Ανάκτηση - Παροχή πληροφοριών σε μορφή άμεσα χρησιμοποιήσιμη ή για περαιτέρω επεξεργασία από άλλες εφαρμογές (Chapple, 2005).
- Διαχείριση - Καταγραφή και έλεγχος των χρηστών, επιβολή της ασφάλειας των δεδομένων, παρακολούθηση της απόδοσης, διατήρηση της ακεραιότητας των δεδομένων Beynon-Davies, 2003).

Το σύστημα βάσης δεδομένων αναφέρεται συλλογικά στο μοντέλο βάσης δεδομένων, το σύστημα διαχείρισης βάσεων δεδομένων και τη βάση δεδομένων (Childs, 1968).

Οι διακομιστές βάσεων δεδομένων είναι συνήθως υπολογιστές πολλαπλών επεξεργαστών, με γενναιόδωρες μνήμες και συστοιχίες δίσκων RAID που χρησιμοποιούνται για σταθερή αποθήκευση (Childs, 1968).

Το RAID χρησιμοποιείται για την ανάκτηση δεδομένων σε περίπτωση αποτυχίας οποιουδήποτε δίσκου. Οι επιταχυντές βάσης δεδομένων υλικού, που συνδέονται με

έναν ή περισσότερους διακομιστές μέσω ενός καναλιού υψηλής ταχύτητας, χρησιμοποιούνται επίσης σε περιβάλλοντα επεξεργασίας συναλλαγών μεγάλου όγκου (Childs, 1968).

Τα DBMS βρίσκονται στην καρδιά των περισσότερων εφαρμογών βάσεων δεδομένων. Τα DBMSs μπορούν να χτιστούν γύρω από έναν προσαρμοσμένο πυρήνα multitasking με ενσωματωμένη υποστήριξη δικτύου.

Δεδομένου ότι τα ΣΔΒΔ αποτελούν σημαντική αγορά, οι πωλητές υπολογιστών και αποθήκευσης λαμβάνουν συχνά υπόψη τις απαιτήσεις ΣΔΒΔ στα δικά τους αναπτυξιακά σχέδια (Chong, Wang, Dang, Snow, 2007).

Οι βάσεις δεδομένων και τα ΣΔΒΔ μπορούν να κατηγοριοποιηθούν σύμφωνα με τα μοντέλα βάσης δεδομένων που υποστηρίζουν, τον τύπο του υπολογιστή στον οποίο εκτελούνται, τη γλώσσα ερωτήματος (s) που χρησιμοποιούνται για την πρόσβαση στη βάση δεδομένων και την εσωτερική τους μηχανική, η οποία επηρεάζει την απόδοση, την επεκτασιμότητα, την ανθεκτικότητα αλλα και την ασφάλεια (Chong, Wang, Dang, Snow, 2007).

ΣΥΣΤΗΜΑΤΑ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

Οι Connolly και Begg ορίζουν το Σύστημα Διαχείρισης Βάσεων Δεδομένων (DBMS) ως «*σύστημα λογισμικού που επιτρέπει στους χρήστες να καθορίζουν, να δημιουργούν, να διατηρούν και να ελέγχουν την πρόσβαση στη βάση δεδομένων*».

Το ακρωνύμιο DBMS διευρύνεται κάποτε για να υποδείξει το υποκείμενο μοντέλο βάσης δεδομένων, με RDBMS για σχεσιακό, OODBMS ή ORDBMS για το αντικείμενο μοντέλο και ORDBMS για Object-Relational (Connolly, Begg, Carolyn, 2014).

Άλλες επεκτάσεις μπορούν να υποδηλώνουν κάποια άλλα χαρακτηριστικά, όπως το DDBMS για συστήματα κατανεμημένης διαχείρισης βάσεων δεδομένων (Connolly, Begg, Carolyn, 2014).

Η λειτουργικότητα που παρέχεται από ένα ΣΔΒΔ μπορεί να διαφέρει πολύ. Η βασική λειτουργικότητα είναι η αποθήκευση, η ανάκτηση και η ενημέρωση των δεδομένων. Ο Codd πρότεινε τις ακόλουθες λειτουργίες και υπηρεσίες που πρέπει να παρέχει ένα ολοκληρωμένο ΣΔΒΣ γενικού σκοπού (Date, 2003):

- Αποθήκευση, ανάκτηση και ενημέρωση δεδομένων
- Κατάλογος χρήστη ή λεξικό δεδομένων προσβάσιμο από τον χρήστη που περιγράφει τα μεταδεδομένα
- Υποστήριξη συναλλαγών και ταυτοχρονισμού
- Οι εγκαταστάσεις για την ανάκτηση της βάσης δεδομένων θα πρέπει αναγκαστικά να καταστραφούν
- Υποστήριξη για την εξουσιοδότηση πρόσβασης και ενημέρωση δεδομένων
- Υποστήριξη πρόσβασης από απομακρυσμένες τοποθεσίες
- Η επιβολή περιορισμών για να εξασφαλιστεί ότι τα δεδομένα στη βάση δεδομένων τηρούν ορισμένους κανόνες

Είναι επίσης γενικά αναμενόμενο ότι το ΣΔΒΔ θα παρέχει ένα σύνολο βοηθητικών προγραμμάτων για τέτοιους σκοπούς, όπως μπορεί να είναι απαραίτητο για την αποτελεσματική διαχείριση της βάσης δεδομένων, συμπεριλαμβανομένων των βοηθητικών προγραμμάτων εισαγωγής, εξαγωγής, παρακολούθησης, ανασυγκρότησης και ανάλυσης (Date, 2003).

Το κεντρικό τμήμα του ΣΔΒΔ που αλληλεπιδρά μεταξύ της βάσης δεδομένων και της διασύνδεσης εφαρμογής που μερικές φορές αναφέρεται ως μηχανισμός βάσης δεδομένων (Date, 2003).

Συχνά τα DBMS θα έχουν παραμέτρους διαμόρφωσης που μπορούν να ρυθμιστούν στατικά αλλα και δυναμικά. Η τάση είναι να ελαχιστοποιηθεί το μέγεθος της χειροκίνητης διαμόρφωσης και για περιπτώσεις όπως οι ενσωματωμένες βάσεις δεδομένων η ανάγκη στόχευσης της μηδενικής διαχείρισης είναι πρωταρχικής σημασίας (Date, 2003).

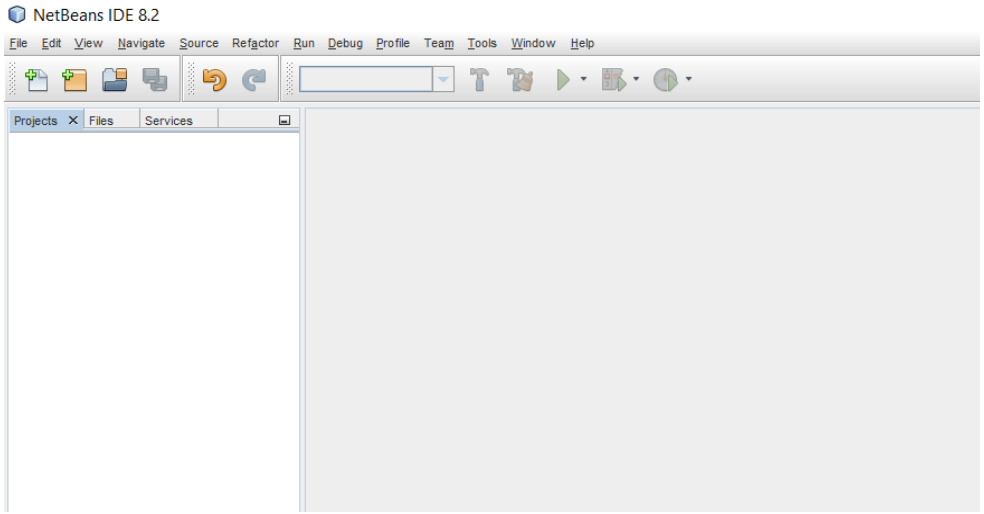
Τα μεγάλα μεγάλα DBMSs των επιχειρήσεων τείνουν να αυξάνονται σε μέγεθος και λειτουργικότητα και μπορούν να έχουν εμπλακεί σε χιλιάδες ανθρώπινα χρόνια αναπτυξιακής προσπάθειας κατά τη διάρκεια της ζωής τους (Date, 2003).

Η αρχιτεκτονική πελάτη-διακομιστή ήταν μια ανάπτυξη όπου η εφαρμογή κατοικούσε σε μια επιφάνεια εργασίας πελάτη και η βάση δεδομένων σε ένα διακομιστή επιτρέποντας τη διανομή της επεξεργασίας.

Η ΕΓΚΑΤΑΣΤΑΣΗ ΤΟΥ NETBEANS 8.2 JDK

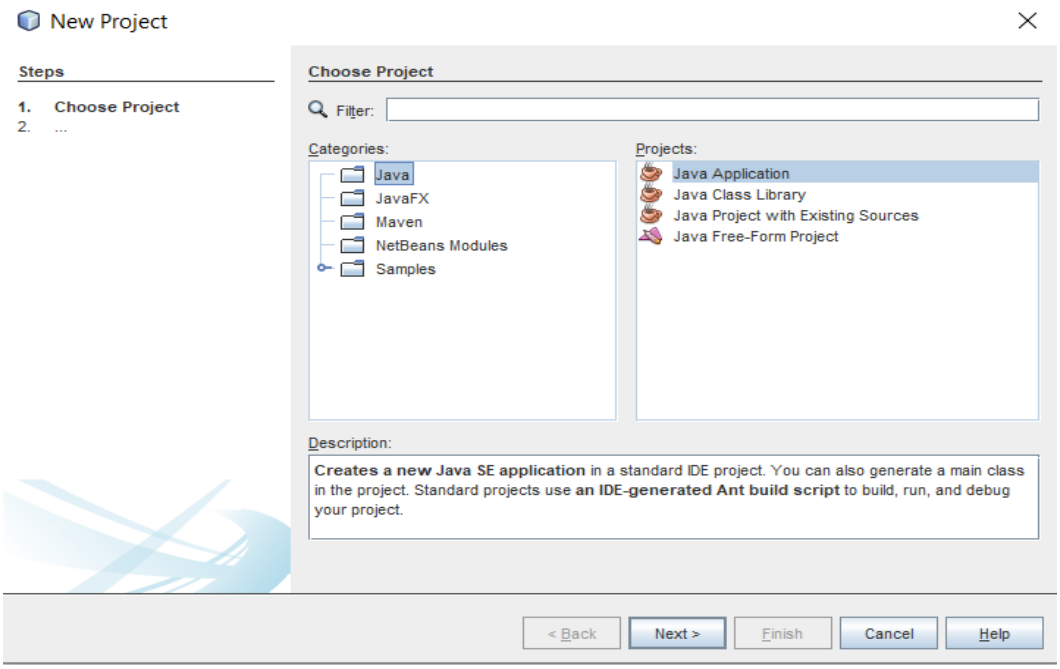
Στην παρούσα φάση της εργασίας θα εξηγήσουμε το πώς μπορεί κάποιος να εγκαταστήσει την πλατφόρμα Netbeans με σκοπό να προγραμματίσει πιο εύκολα και αποδοτικά με την γλώσσα προγραμματισμού Java. Αρχικά πρέπει να μεταβεί στην παρακάτω σελίδα και να κατεβάσει το ανάλογο πακέτο, σύμφωνα με το λειτουργικό σύστημα που αυτός χρησιμοποιεί.

Αφού προχωρήσει στην εγκατάσταση της εφαρμογής, μόλις ανοίξει την εφαρμογή θα παρατηρήσει το παρακάτω κύριο μενού επιλογών(**Εικονα: 1**)



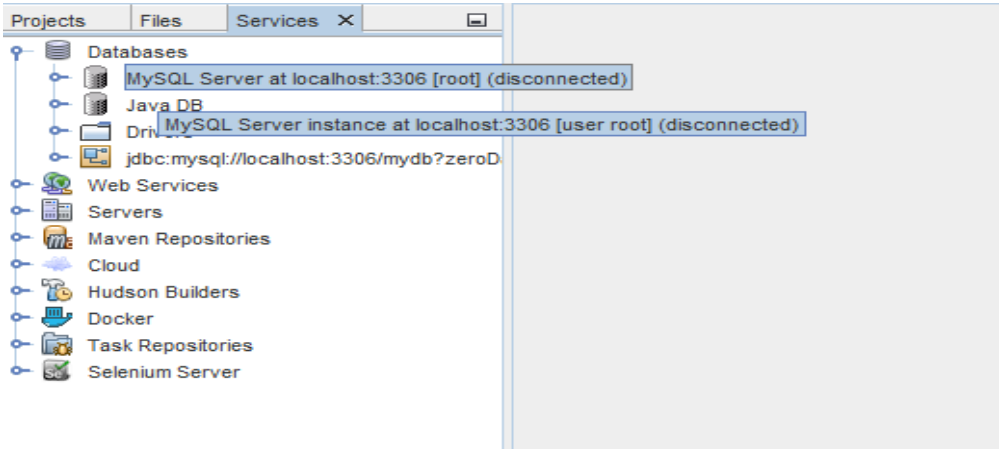
Εικόνα 1: Κεντρικό μενού του Netbeans

Όπως παρατηρούμε διαθέτει μια κεντρική μπάρα οπου μπορεί κάποιος να επιλέξει πχ να τρέξει την εφαρμογή κ.α Επίσης διαθέτει την μπάρα Edit μέσω της οποίας μπορεί πολύ ευκολα να κανει copy, paste και αλλες λειτουργίες στην εφαρμογή του. Μέσω του File, μπορει κάποιος να δημιουργήσει μια νέα εφαρμογή, αναλόγως τον τύπου που ζητάει (**Εικόνα: 2**)



Εικόνα 2: Η δημιουργία νέας εφαρμογής

Φυσικά μπορεί να συνδέσει την εφαρμογή του με οποιαδήποτε είδους εφαρμογή εγκαθιστώντας τον κατάλληλο driver. Εφόσον ειναι live η βαση δεδομένων του μπορεί στη συνέχεια να την συνδέσει με το Netbeans για να μπορεί παράλληλα να βλέπει τις αλλαγές στην βάση δεδομενων που γίνονται διαμέσου του Netbeans(**Εικόνα: 3**)



Εικόνα 3: Συνδεση βασης δεδομενων με Netbeans

Η ΕΓΚΑΤΑΣΤΑΣΗ ΤΗΣ SQL COMMUNITY SERVER

Για να μπορέσουμε να δημιουργήσουμε ενα σύστημα βάσης δεδομένων κατάλληλο για την αεροπορική μας εταιρεία χρησιμοποιήσαμε το SQL Community Server το οποίο είναι γρήγορο, ευέλικτο και δωρεάν.

Επομένως για να κρατάμε κάπου τα δεδομένα μας, είναι απαραίτητο να χρησιμοποιούμε μια βάση δεδομένων είτε αυτό αφορά την δική μας εφαρμογή είτε οποιαδήποτε άλλη.

Για να κατεβάσουμε το παραπάνω πακέτο λογισμικού μεταβήκαμε στην παραπάνω σελίδα. Στο μέσον της εγκατάστασης μας ζητά εάν θα κατεβάσουμε το full πακέτο τύπου client συν σέρβερ , ή απλά το πακέτο για development purposes.

Επίσης, μας ζητάει να θέσουμε έναν κωδικό για την βάση, με σκοπό να μην είναι προσβάσιμη από τρίτους. Μόλις ανοίξουμε την εφαρμογή μας θα εμφανιστεί το παρακάτω main menu (**Εικόνα: 4**)

Εικόνα 4: Κυριο μενού της MySQL WorkBench

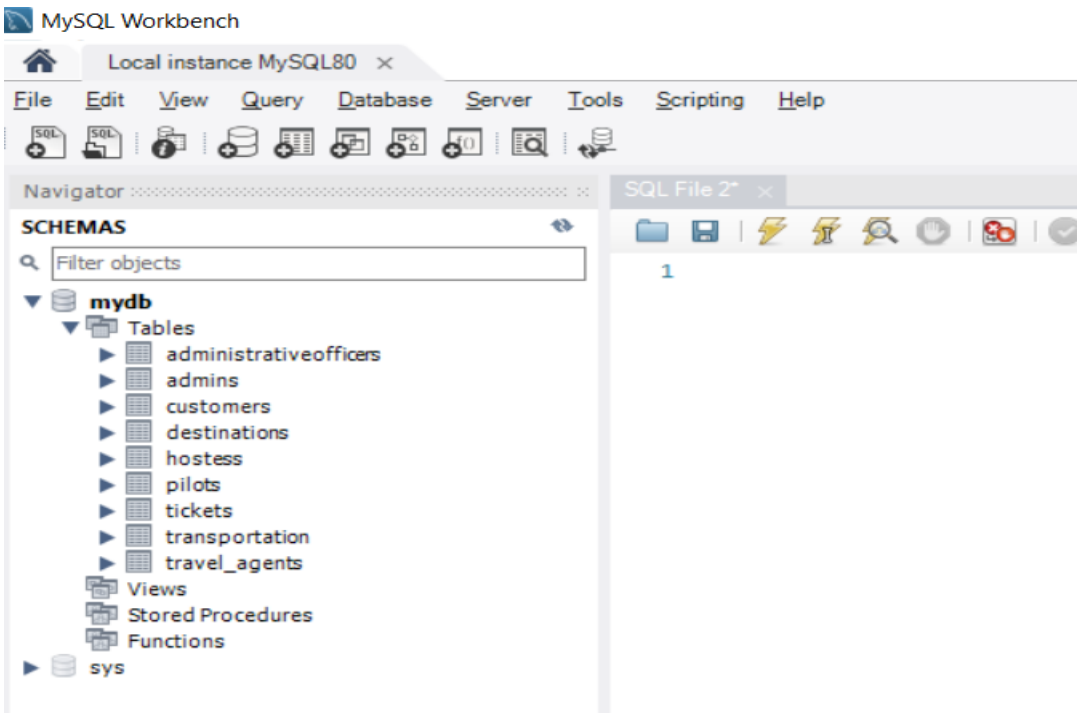
Μέσω του παραπάνω μενού μπορούμε να δημιουργήσουμε ένα νέο σχεσιακό σχήμα, μια νέα βαση δεδομένων κ.α. Το MySql Connections ουσιαστικά μας δείχνει τις βάσεις δεδομένων που έχουμε δημιουργήσει ενώ όταν πατάμε να συνδεθούμε σε κάποια απο αυτές αναγκαστικά θα δώσουμε τον κωδικό της βάσης δεδομένων της εκάστωτε βασης.

Στο MySQL WorkBench λειτουργούν κανονικά οι εντολές SQL για δημιουργία νέου πίνακα, διαγραφή πίνακα, εισαγωγή στηλών, διαγραφή στηλών και άλλες λειτουργίες.

ΠΕΙΡΑΜΑΤΙΚΟ ΜΕΡΟΣ: Η ΕΦΑΡΜΟΓΗΣ ΑΕΡΟΠΟΡΙΚΗΣ ΕΤΑΙΡΕΙΑΣ

ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΠΟΥ ΔΗΜΙΟΥΡΓΗΣΑΜΕ

Μόλις ανοίξουμε το MySQL Workbench θα πατήσουμε επάνω στην βάση μας, θα δώσουμε τον κατάλληλο κωδικό και θα μπούμε στο κύριο σημείο της βάσης μας. Η βάση μας έχει ως εξής (**Εικόνα: 5**)



Εικόνα 5: Η βαση δεδομένων μας

Στην βάση μας δημιουργήσαμε τα tables που φαίνονται παραπάνω. Ενώ κάθε table έχει διαφορετική μορφή, λόγω των διαφορετικών πεδίων που διαθέτει το κάθε ένα.

Για παράδειγμα το table customers διαθέτει τα πεδία(**Εικόνα: 6**)

Table: customers

Columns:

customer_id	int(6) UN AI PK
firstname	varchar(30)
lastname	varchar(30)
tel	varchar(50)
street_name	varchar(50)
street_number	varchar(50)
town	varchar(50)
username	varchar(50)
customer_password	varchar(50)
created_at	timestamp

Εικόνα 6: Τα πεδία του customers

Από την άλλη πλευρά, το table travel_agents διαθέτει τα πεδία που φαίνονται στην παρακάτω εικόνα (**Εικόνα: 7)**

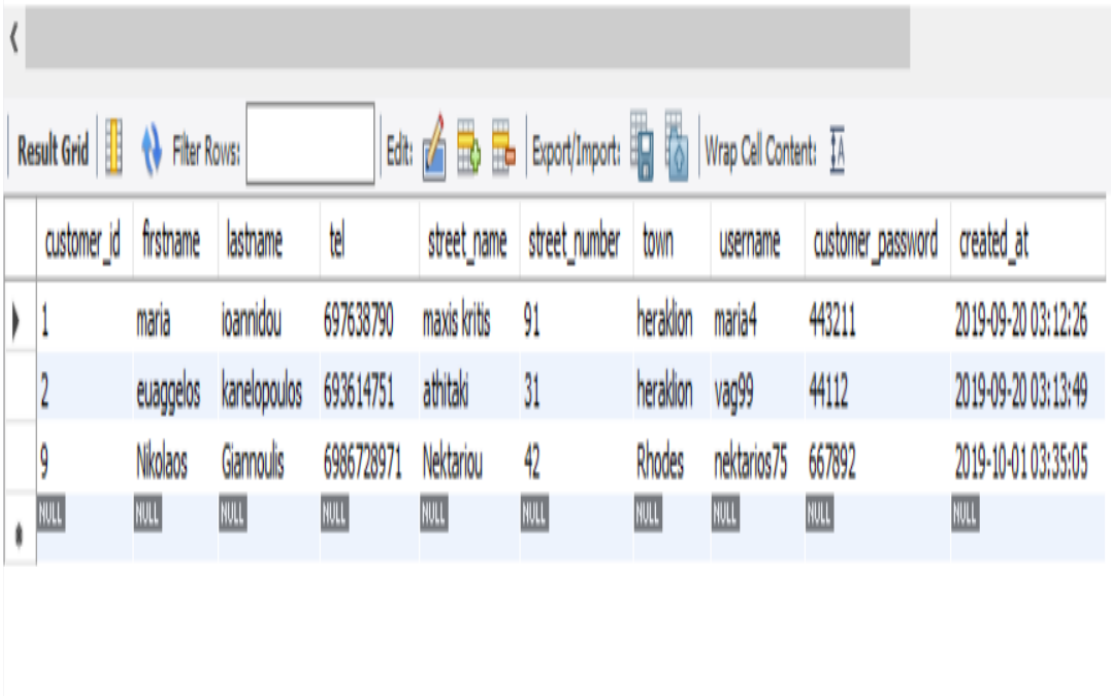
Εικόνα 7: Τε περιεχόμενα του travel_agents

Όσο αφορά τα περιεχόμενα των admins μπορούμε να πούμε ότι (**Εικόνα : 8**)

Εικόνα 8: Τα περιεχόμενα του table , admins

Να τονίσουμε ότι το κενό πεδίο στην αρχικό μενού είναι για να μπορούμε να γράφουμε τα SQL queries και όταν στη συνέχεια πατάμε το σημαδάκι τύπου «κεραυνός» που βρίσκεται στην μπάρα επιλογών(οριζόντια) εμφανίζεται στο ενδιάμεσο το αποτέλεσμα του SQL query που δώσαμε στο συγκεκριμένο table που επιλέξαμε.

Πιο συγκεκριμένα εάν γράψουμε το query select * from customers τότε θα εμφανιστεί το εξής αποτέλεσμα στην οθόνη μας (**Εικόνα: 9**)



customer_id	firstname	lastname	tel	street_name	street_number	town	username	customer_password	created_at
1	maria	ioannidou	697638790	maxis kritis	91	heraklion	maria4	443211	2019-09-20 03:12:26
2	euaggelos	kanelopoulos	693614751	athitaki	31	heraklion	vag99	44112	2019-09-20 03:13:49
9	Nikolaos	Giannoulis	6986728971	Nektariou	42	Rhodes	nektarios75	667892	2019-10-01 03:35:05
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Εικόνα 9: SQL query για την εμφάνιση των customer

Στις παρακάτω εικόνες θα δούμε συγκεκριμένα ολα τα tables της βάσης δεδομένων που χρησιμοποιήσαμε, με σκοπό να έχουμε μια άποψη για τις συνδέσεις μεταξύ αυτών.

Πιο συγκεκριμένα στην(**Εικονα : 10**) βλέπουμε τους administrativeofficers

Εικόνα 10: administrativeofficers table

Στη συνέχεια στην (**Εικόνα : 11**) τους admins

Εικόνα 11: admins table

Επιπρόσθετα στην (**Εικόνα : 12**) βλέπουμε το bookings

Εικόνα 12: bookings table

Πιο συγκεκριμένα στην (**Εικόνα : 13**) παρατηρούμε

Εικόνα 13: customers table

Πιο συγκεκριμένα στην (**Εικόνα : 14**) βλέπουμε το destinations

Εικόνα 14:destinations table

Πιο αναλυτικά στην (**Εικονα : 15**) παρατηρούμε το table final_bookings που διαθέτει ακριβώς τα ίδια πεδία με το table bookings.

Εικόνα 15:final_bookings table

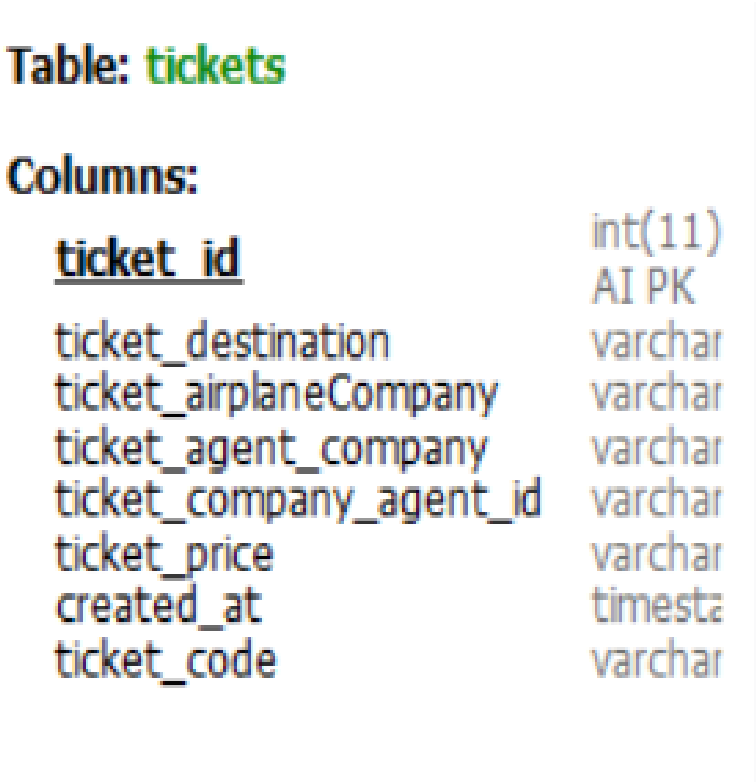
Πιο συγκεκριμένα στην (**Εικόνα : 16**) βλέπουμε το hostess

Εικόνα 16:hostess table

Πιο συγκεκριμένα στην (**Εικόνα : 17**) βλέπουμε το pilots

Εικόνα 17:pilots table

Πιο συγκεκριμένα στην (**Εικονα : 18**) βλέπουμε το tickets



Εικόνα 18: tickets table

Πιο συγκεκριμένα στην (**Εικόνα : 19**) βλέπουμε το transportation

Εικόνα 19:transportation table

Πιο συγκεκριμένα στην (**Εικόνα : 20**) βλέπουμε το travel_agents

Εικόνα 20: travel_agents table

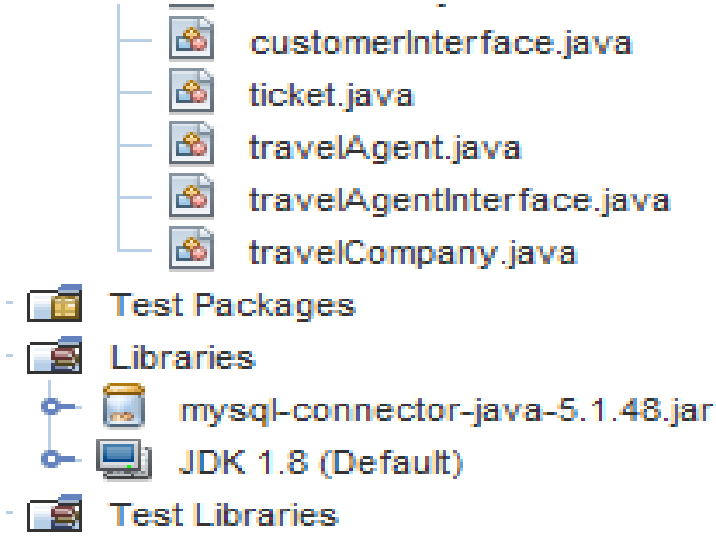
ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΜΑΣ

ΣΥΝΔΕΣΗ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ΜΕ ΤΟ NETBEANS

Εφόσον έχουμε εγκαταστήσει κανονικά το Netbeans, διαθέτουμε την τελευταία έκδοση της Java αλλά και προεγκαταστήσει επίσης το MySQL Workbench τότε είμαστε έτοιμοι να συνδέσουμε την βάση δεδομένων μας με το πρόγραμμα Netbeans.

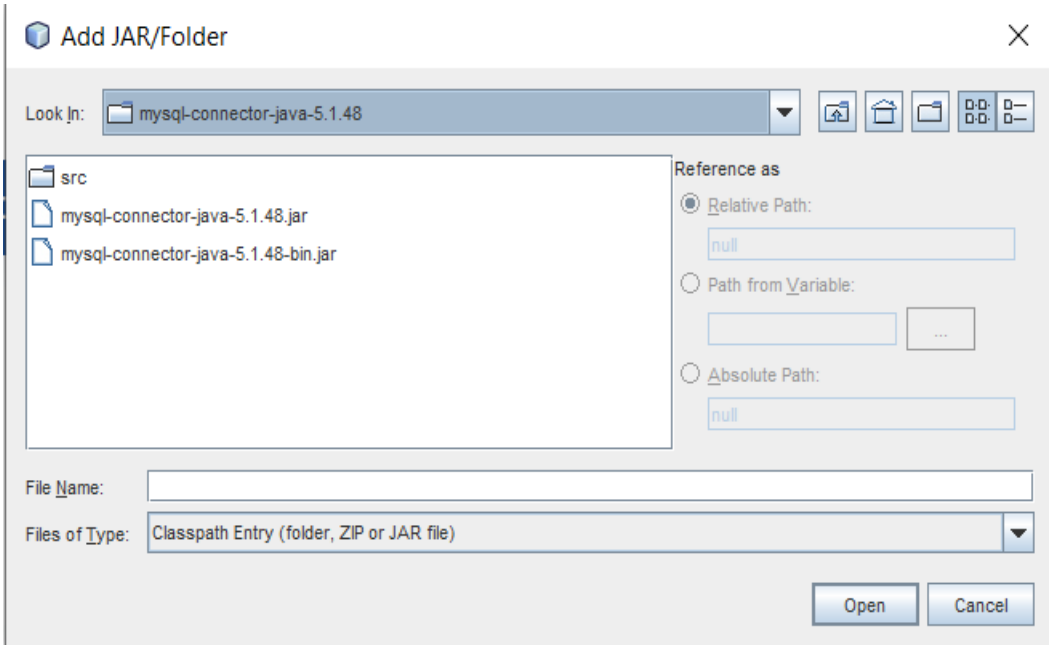
Αρχικά ανοίγουμε το πρότζεκτ το οποίο έχουμε δημιουργήσει

Πηγαίνουμε στην ενότητα libraries (**Εικόνα: 21**)



Εικόνα 21: project_libraries

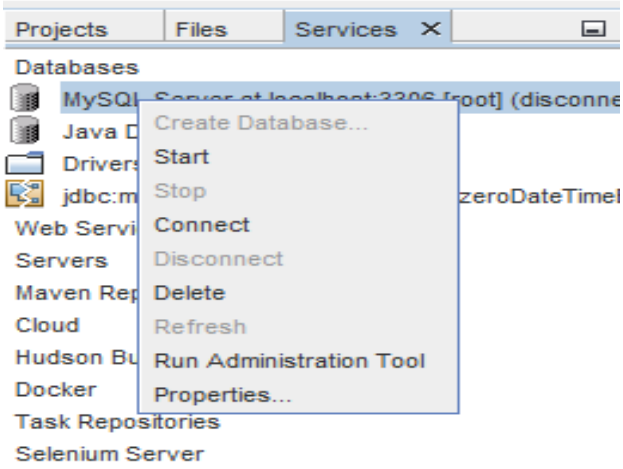
Κατόπιν πατάμε δεξί κλικ και επιλέγουμε ADD/JAR Folder και μας ανοίγει το παρακάτω πάνελ (**Εικόνα: 22**)



Εικόνα 22: add mysql-connector

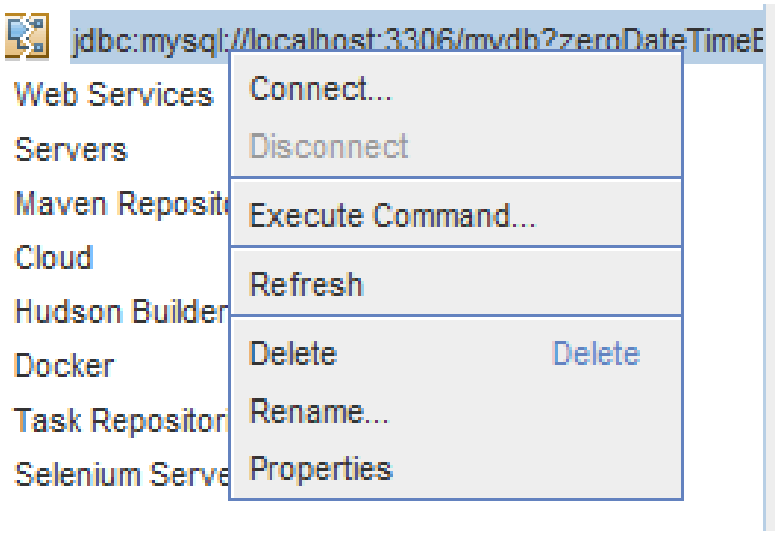
Δηλαδή εισάγουμε τον κατάλληλο connector για την version του Netbeans και windows που χρησιμοποιούμε.

Στη συνέχεια πατάμε στην επιλογή services του πρότζεκτ μας (**Εικόνα: 23**)



Εικόνα 23: connect to a database

Τοποθετούμε τα στοιχεία της βάσης δεδομένων μας και στη συνέχεια έχουμε συνδεθεί στην βάση. Αυτό εάν γίνει σωστά θα φανεί κάπως έτσι (**Εικόνα: 24**)



Εικόνα 24:Μορφη επιτυχούς σύνδεσης με βάση δεδομένων

MAIN

Στην main() προσπαθήσαμε να τοποθετήσουμε κατάλληλο κώδικα που να μπορεί να δημιουργεί ενα κατάλληλο JFrame ώστε ο εκάστωτε χρήστης να βάζει username και password και πατώντας ενα button τυπου JButton να μπορεί να συνδεθεί κατάλληλα στο δικό του interface, αναλόγως των στοιχείων που έδωσε ο χρήστης.

Επομένως μπορεί να είναι είτε admin, ειτε customer αλλά είτε travel agent. Επιπρόσθετα στο ίδιο JFrame δώσαμε κατάλληλο κώδικα ώστε να μπορεί να γίνεται εγγραφή customer στο σύστημα.

Πιο συγκεκριμένα, ο κώδικας πρέπει κατάλληλα να ανοίγει την βάση δεδομένων μας, να δίνεται η πρόσβαση με τα κατάλληλα queries στον πίνακα των customers και να εγγράφεται αφού πρώτα ελεγχθεί, εαν υπάρχει όμοιος πελάτης με κάποιο συγκεκριμένο αναγνωριστικό.

Εάν υπάρχει δηλαδή ο ίδιος πελάτης στο σύστημα τότε η εγγραφή δεν γίνεται κανονικά. Έτσι εξασφαλίζουμε την μοναδικότητα των στοιχείων και κατ' επέκταση των πελατών της βάσης δεδομένων μας.

Αρχικά εισάγουμε τα κατάλληλα J Imports στην main() κλάση μας (**Εικόνα: 25**)

```
package gerakianakis_thesis_2019;

import java.awt.BorderLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;
```

Εικόνα 25: basic J Imports

Στη συνέχεια δημιουργούμε ενα κατάλληλο JFrame με τα κατάλληλα JLabels αλλά και JTextfields (και JPasswordFields). Αυτά θα ενσωματωθούν σε ένα κατάλληλο JPanel που προηγουμένως έχει ενσωματωθεί στο JFrame μας. Επιλέγουμε ως Layout το BorderLayout και GridLayout με σκοπό να χωρίσουμε το JFrame σε κατάλληλα μέρη.

Όπως βλέπουμε στην (**Εικόνα: 26**) θα χρειαστούμε δύο JLabels, ενα JTextField και ενα JPasswordField για την είσοδο του χρήστη στο σύστημα. Επιπρόσθετα χρειαζόμαστε και ένα κουμπί για το register αλλά και το login.

```
JFrame frame = new JFrame();
frame.setSize(300, 450);
JPanel panel = new JPanel(new GridLayout(6, 1));
frame.add(panel, BorderLayout.CENTER);
//login & register form
JLabel username = new JLabel("Username:");
JLabel password = new JLabel("Password:");
JTextField field1 = new JTextField(30);
JPasswordField field2 = new JPasswordField(30);
JButton login = new JButton("Login");
JButton register = new JButton("Register");
//add everything to panel
panel.add(username);
panel.add(field1);
panel.add(password);
panel.add(field2);
panel.add(login);
panel.add(register);
//ACTIONLISTENERS FOR REGISTER & LOGIN BUTTONS
//login forms for travel agents, admins, customers
login.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent ae) {

        try {
            String username = field1.getText();
            String password = field2.getText();
```

Εικόνα 26: Login & Register code

Επίσης, για να τραβήξουμε τις τιμές απο το event του JButton χρησιμοποιούμε ActionListener και μέσα σε αυτόν ορίζουμε κατάλληλα Strings και τραβάμε τις τιμές της φόρμας με την συνάρτηση .getText()

```
try {
    String username = field1.getText();
    String password = field2.getText();
    //check our database -which user is logged in?
    Class.forName("com.mysql.jdbc.Driver");
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
        Statement stmt1 = con.createStatement();
        Statement stmt2 = con.createStatement();
        Statement stmt3 = con.createStatement();
        ResultSet rs1 = stmt1.executeQuery("select * from admins");
        ResultSet rs2 = stmt2.executeQuery("select * from customers");
        ResultSet rs3 = stmt3.executeQuery("select * from travel_agents");
        //admin checking
        while (rs1.next()) {
            if (username.equals(rs1.getString(4)) && password.equals(rs1.getString(5))) {
                System.out.println("YOU LOGGED AS ADMIN :" + rs1.getString(4));
                //go to admin interface class
                JButton b1 = new JButton();
                b1.addActionListener(new adminInterface());
                con.close();

            }
```

Εικόνα 27: database connection plus admin checking

Στην παραπάνω εικόνα ανοίγουμε την βάση δεδομένων, επιλέγουμε τα πάντα από κάθε κατάλληλο table των χρηστών και μέσω κατάλληλης while κάνουμε αναζήτηση στον εκάστωτε πίνακα για να δούμε πχ εάν υπάρχει το ίδιο $4^{ο}$ και $5^{ο}$ πεδίο του table admin όμοιο με τα στοιχεία της φόρμας που δώσαμε.

Δηλαδή εάν υπάρχει ίδιο username και password ταυτόχρονα (χρησιμοποιούμε την &&) τότε ο χρήστης δεν θα μεταβεί στο δικό του interface που στην προκείμενη περίπτωση για τον admin ειναι το adminInterface().

Για να μεταβούμε πιο έυκολα χρησιμοποιούμε ενα κουμπί που δεν εμφανίζεται πουθενά, απλά γίνεται αυτομάτως event απο μόνο του, και μέσω αυτού θα κληθεί ένας κατάλληλος ActionListener για να στείλει τον χρήστη πχ admin στο interface του.

Παρομοίως στην (**Εικόνα: 28)** θα δούμε οτι το ίδιο ισχύει για τον πελάτη.

```
//customer checking
while (rs2.next()) {
    if (username.equals(rs2.getString(8)) && password.equals(rs2.getString(9))) {
        System.out.println("YOU LOGGED AS CUSTOMER :" + rs2.getString(8));
        //go to customer interface class
        JButton b2 = new JButton();
        //We wanted also, to pass customer's username via ActionListener in order to use it into customerInterface class
        String username_s = rs2.getString(8);
        b2.addActionListener(new customerInterface(username_s));
        con.close();
    }
}
//travel agent checking
while (rs3.next()) {
    if (username.equals(rs3.getString(7)) && password.equals(rs3.getString(8))) {
        System.out.println("YOU LOGGED AS TRAVEL AGENT :" + rs3.getString(7));
        //go to travel agent interface class
        JButton b3 = new JButton();
        b3.addActionListener(new travelAgentInterface());
        con.close();
    }
```

Εικόνα 28: customer/agent checking

Όσο αφορά την εγγραφή χρήστη θα χρειαστούμε ενα κατάλληλο JFrame με τα κατάλληλα πεδία που αναλόγως τον χρήστη που θα εγγραφεί είτε αυτός είναι customer είτε travel agent με δύο κατάλληλα κουμπία θα μεταβούμε στις κλάσεις εγγραφής αυτών (**Εικόνα: 29)**

```
//ActionListeners for register buttons
customer.addActionListener(new addCustomer());
travel_agent.addActionListener(new addTravelAgent());
exit.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent ae) {
        frame.dispose();
    }
});
frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
frame.setVisible(true);
frame.setResizable(false);
```

Εικόνα 29: μετάβαση στην εγγραφή χρηστών

Επίσης η συνάρτηση frame.dispose() ουσιαστικά όταν γίνει ένα event μέσω του κουμπιού που οδηγεί σε αυτή, κανει dispose το JFrame αντί να κλείσει ολόκληρη η εφαρμογή που εμείς τρέχουμε αυτή τη στιγμή.

Για παράδειγμα για την εγγραφή ενός customer πάλι δημιουργούμε το κατάλλλο frame που θα διαθέτει τα πεδία που έχουμε ορίσει ότι ένας πελάτης διαθέτει, που είναι όμοια με αυτά που βρίσκονται και στην βάση δεδομένων στον πίνακα customers (**Εικόνα: 30**).

```
JLabel firstname = new JLabel("FIRSTNAME:");
JLabel lastname = new JLabel("LASTNAME:");
JLabel tel = new JLabel("TEL:");
JLabel street_name = new JLabel("STREET NAME:");
JLabel street_number = new JLabel("STREET NUMBER:");
JLabel town = new JLabel("TOWN:");
JLabel username = new JLabel("USERNAME:");
JLabel password = new JLabel("PASSWORD:");

JTextField firstname_field = new JTextField(30);
JTextField lastname_field = new JTextField(30);
JTextField tel_field = new JTextField(30);
JTextField streetName_field = new JTextField(30);
JTextField streetNumber_field = new JTextField(30);
JTextField town_field = new JTextField(30);
JTextField username_field = new JTextField(30);
JPasswordField password_field = new JPasswordField(30);
JButton register = new JButton("REGISTER");

//add everything to panel
panel.add(firstname);
panel.add(firstname_field);
panel.add(lastname);
panel.add(lastname_field);
panel.add(tel);
panel.add(tel_field);
panel.add(street_name);
panel.add(streetName_field);
panel.add(street_number);
panel.add(streetNumber_field);
panel.add(town);
panel.add(town_field);
panel.add(username);
panel.add(username_field);
panel.add(password);
panel.add(password_field);
panel.add(register);
```

Εικόνα 30: customer's form

Η εγγραφή θα γίνει ανοίγοντας κατάλληλα την βάση δεδομένων, χρησιμοποιώντας MySQL γράφουμε το κατάλληλο query για την εγγραφή στον πίνακα customers και μέσω κατάλληλων PreparedStatements, ResultSet και while, γράφουμε την εγγραφή στον πίνακα μας (**Εικόνα:31**).

```
try {
    Class.forName("com.mysql.jdbc.Driver");
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("select * from customers");
        String query = "INSERT INTO customers (firstname,lastname,tel,street_name,street_number,town,username,customer_password)" + "VALUES (?,?,?,?,?,?,?,?)";
        PreparedStatement preparedStmt = con.prepareStatement(query);

        PreparedStatement pStmnt = null;
        String preQueryStatement = "select  * from customers where customers.username = ?";
        pStmnt = con.prepareStatement(preQueryStatement);
        pStmnt.setString(1, username);
        ResultSet rs1 = pStmnt.executeQuery();

        //if customer does not exists then add to customer's database
        while (!rs1.next()) {
            preparedStmt.setString(1, firstname);
            preparedStmt.setString(2, lastname);
            preparedStmt.setString(3, tel);
            preparedStmt.setString(4, street_name);
            preparedStmt.setString(5, street_number);
            preparedStmt.setString(6, town);
            preparedStmt.setString(7, username);
            preparedStmt.setString(8, customer_password);
            preparedStmt.execute();
            con.close();
            System.out.println("CUSTOMER REGISTERED SUCCESSFULLY");

        }

    }

} catch (ClassNotFoundException | SQLException ex) {
```

Εικόνα 31: εγγραφή customer στην βάση δεδομένων μας

Με παρόμοιο τρόπο γράφεται και ένας travel agent όπως φαίνεται παρακάτω (**Εικόνα: 32**)

```
try {
    Class.forName("com.mysql.jdbc.Driver");
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("select * from travel_agents");
        String query = "INSERT INTO travel_agents (fname,lname,AT_number,street_name,street_number,agent_username,agent_password)" + "VALUES (?,?,?,?,?,?,?)";
        PreparedStatement preparedStmt = con.prepareStatement(query);

        PreparedStatement pStmnt = null;
        String preQueryStatement = "select  * from travel_agents where travel_agents.agent_username = ?";
        pStmnt = con.prepareStatement(preQueryStatement);
        pStmnt.setString(1, agent_username  );
        ResultSet rs1 = pStmnt.executeQuery();

        //if travel agent does not exists then add to travel agents database
        while (!rs1.next()) {
            preparedStmt.setString(1, fname);
            preparedStmt.setString(2, lname);
            preparedStmt.setString(3, AT_number);
            preparedStmt.setString(4, street_name);
            preparedStmt.setString(5, street_number);
            preparedStmt.setString(6, agent_username);
            preparedStmt.setString(7, agent_password);
            preparedStmt.execute();
            con.close();
```

Εικόνα 32:εγγραφή travel agent στην βάση δεδομένων μας

ADMIN SIDE

Ένας admin μπορεί να δημιουργεί μια μεταφορά, να την εκτυπώνει ενώ επιπρόσθετα να προσθέτει κατάλληλους υπαλλήλους στο σύστημα. Αρχικά δημιουργούμε ένα κατάλληλο JFrame, JPanel, κατάλληλα JButtons (Εικόνα: 33)

```
JFrame frame = new JFrame();
frame.setSize(300, 450);
JPanel panel = new JPanel(new GridLayout(4, 1));
frame.add(panel, BorderLayout.CENTER);

JButton create_transportation = new JButton("CREATE TRANSPORTATION");
JButton print_transportations = new JButton("PRINT TRANSPORTATION");
JButton add_employees = new JButton("ADD EMPLOYEES");
JButton exit_b = new JButton("EXIT");

//add everything to panel
panel.add(create_transportation);
panel.add(print_transportations);
panel.add(add_employees);
panel.add(exit_b);
```

Εικόνα 33: Admin basic JFrame

Αναλόγως το event του κουμπιού θα μεταβούμε σε κατάλληλο ActionListener(). Πιο συγκεκριμένα ο ActionListener που αφορά την προσθήκη των χρηστών διαθέτει ένα JFrame, JPanel και 3 κουμπιά μιας και οι υπάλληλοι είναι τρείς στο σύνολο(pilots, hostess, officers).

Επομένως χρειαζόμαστε μέσα το ίδιο JFrame , τρία κουμπία άρα και τρείς ξεχωριστούς actionListeners που ο κάθε ένας θα γράφει στην βάση τον κατάλληλο υπάλληλο που εμείς επιλέξαμε ως admin να εισάγουμε (**Εικόνα: 34**).

```
JFrame add_employees_frame = new JFrame();
add_employees_frame.setSize(300, 450);
JPanel add_employees_panel = new JPanel(new GridLayout(3, 1));
add_employees_frame.add(add_employees_panel, BorderLayout.CENTER);
JButton add_pilots = new JButton("ADD PILOT");
JButton add_hostess = new JButton("ADD HOSTESS");
JButton add_officers = new JButton("ADD OFFICER");

//add everything to panel
add_employees_panel.add(add_pilots);
add_employees_panel.add(add_hostess);
add_employees_panel.add(add_officers);
//actionlisteners

//adding pilots
add_pilots.addActionListener(new ActionListener() {...112 lines });

//adding hostess
add_hostess.addActionListener(new ActionListener() {...112 lines });
//adding officers
add_officers.addActionListener(new ActionListener() {...119 lines });

add_employees_frame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
add_employees_frame.setVisible(true);
add_employees_frame.setResizable(false);
```

Εικόνα 34: Εισαγωγή υπαλλήλου ξεχωριστά στην βάση δεδομένων

Για παράδειγμα για να προσθέσουμε πιλότους στο σύστημα δημιουργούμε πάλι ένα JFrame με τα κατάλληλα JLabels και JTextFields για να πάρουμε τις τιμές της φόρμας που στη συνέχεια θα τα περάσουμε στη βάση δεδομένων μας στο table pilots κτλ (**Εικόνα: 35**).

```
//adding pilots
add_pilots.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent ae) {
        JFrame pilot_frame = new JFrame();
        pilot_frame.setTitle("PILOT REGISTER");
        pilot_frame.setSize(600, 480);
        JPanel pilot_panel = new JPanel(new GridLayout(19, 1));
        pilot_frame.add(pilot_panel, BorderLayout.CENTER);

        JLabel fname_pilot = new JLabel("FIRSTNAME:");
        JLabel lname_pilot = new JLabel("LASTNAME:");
        JLabel id_pilot = new JLabel("ID NUMBER:");
        JLabel street_name_pilot = new JLabel("STREET NAME:");
        JLabel street_number_pilot = new JLabel("STREET NUMBER:");
        JLabel town_pilot = new JLabel("TOWN:");
        JLabel tel_pilot = new JLabel("TELEPHONE");
        JLabel birth_pilot = new JLabel("DATE OF BIRTH");
        JLabel flight_hours_pilot = new JLabel("FLIGHT HOURS");

        JTextField fname_pilot_f = new JTextField(30);
        JTextField lname_pilot_f = new JTextField(30);
        JTextField id_pilot_f = new JTextField(30);
        JTextField street_name_pilot_f = new JTextField(30);
        JTextField street_number_pilot_f = new JTextField(30);
        JTextField town_pilot_f = new JTextField(30);
        JTextField tel_pilot_f = new JTextField(30);
        JTextField birth_pilot_f = new JTextField(30);
        JTextField flight_hours_pilot_f = new JTextField(30);
        JButton pilot_register = new JButton("REGISTER");

        //add everything to panel
```

Εικόνα 35: φόρμα των στοιχείων του pilot

Οπότε με κατάλληλο κώδικα ανοίγουμε την βάση και προσθέτουμε την εγγραφή μας (**Εικόνα : 36**).

```
//storing into pilots database...
try {
    Class.forName("com.mysql.jdbc.Driver");
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root"))
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("select * from pilots");
        String query = "INSERT INTO pilots (fname,lname,id_number,address_name,address_number,town,telephone,date_birth,flight_hours)" + "
        PreparedStatement preparedStmt = con.prepareStatement(query);

        PreparedStatement pStmnt = null;
        String preQueryStatement = "select  * from pilots where pilots.id_number = ?";
        pStmnt = con.prepareStatement(preQueryStatement);
        pStmnt.setString(1, pilot_id);
        ResultSet rs1 = pStmnt.executeQuery();

        //if customer does not exists then add to customer's database
        while (!rs1.next()) {
            preparedStmt.setString(1, pilot_firstname);
            preparedStmt.setString(2, pilot_surname);
            preparedStmt.setString(3, pilot_id);
            preparedStmt.setString(4, pilot_street_name);
            preparedStmt.setString(5, pilot_street_number);
            preparedStmt.setString(6, pilot_town);
            preparedStmt.setString(7, pilot_tel);
            preparedStmt.setString(8, pilot_birth);
            preparedStmt.setString(9, pilot_flight_hours);
            preparedStmt.execute();
            con.close();
            System.out.println("PILOT REGISTERED SUCCESSFULLY");
```

Εικόνα 36:εισαγωγη pilot στην βαση δεδομενων

Για να κάνουμε μια μεταφορά θα χρειαστούμε έναν κυβερνήτη, έναν συγκυβερνήτη και τέσσερις αεροσυνοδούς. Ο πίνακας των μεταφορών παρουσιάστηκε σε παραπάνω ενότητα. Ουσιαστικά θα συνδεθούμε όπως πάντα στην βάση δεδομένων μας ως admin.

Στη συνέχεια θα επιλέξουμε create_transporation και θα γράψουμε μια εγγραφή στον πίνακα transportation. Όμως μιας και πρέπει σε κάθε μεταφορά να υπάρχουν δύο πιλότοι και τέσσερις αεροσυνοδοί θα πρέπει ταυτόχρονα να αφαιρέσουμε (οπως περίπου κάναμε και στην πλευρά του customer, την διαδικασία το create booking) τους δύο πιλότους από τον πίνακα pilots και τις τέσσερις αεροσυνοδούς που εμείς θα επιλέξουμε από τον πίνακα hostess.

Δηλαδή αφότου γίνει η εγγραφή της μεταφοράς στον πίνακα transportation, θα ανοίξουμε δύο φορές την βάση. Μια φορά για τον πίνακα pilots και μια φορά για τον πίνακα hostess.

Όταν ανοίξουμε τον πίνακα pilots πχ select * from pilots; Ευθύς αμέσως θα πρέπει να δημιουργήσουμε ένα κενό PreparedStatement, στη συνέχεια να θέσουμε ένα κατάλληλο string που θα εμπεριέχει το MySQL query διαγραφής πχ με βάση τον αριθμό ταυτότητας ή αλλο αναγνωριστικό που περιέχει ο πινακας pilots, τέλος θα τρέξουμε executeQuery(); και θα σβηστούν ο πιλότος που θέλουμε.

Με κατάλληλο τρόπο μπορούμε να αφαιρέσουμε δύο πιλότους εάν για παράδειγμα θέσουμε δύο διαφορετικά strings που το κάθε ένα θα εμπεριέχει το αναγνωριστικό του πίνακα pilots (που θέσαμε στην φόρμα δημιουργίας μιας μεταφοράς) και στη συνέχεια με δύο διαφορετικά executeQuery(); , να καταφέρουμε να σβήσουμε τους δύο πιλότους από την βάση δεδομένων μας.

Η διαγραφή των πιλότων σημαίνει ότι απλά οι πιλότοι «βρίσκονται ήδη σε κάποια μεταφορά, και σε κάποιο δρομολόγιο την παρούσα χρονική στιγμή»

Όσο αφορά τις αεροσυνοδούς(hostess), με παρόμοιο τρόπο μπορούμε να τρέξουμε τέσσερα διαφορετικά είδη PreparedStatements() ή ενα συμπυκνωμένο, για να αφαιρέσουμε τρέχοντας το executeQuery() να σβήσουμε τις τέσερις αεροσυνοδούς.

Όσο αφορά το print των μεταφορών είναι ακριβώς ίδιο με το print οποιουδήποτε άλλου πίνακα που περιγράψαμε παραπάνω. Πιο συγκεκριμένα, χρειαζόμαστε ένα κατάλληλο JFrame, ενα JScrollPane (σε περίπτωση που έχουμε πολλές εγγραφές οι οριζόντες και κατακόρυφες μπάρες κύλισης ειναι απαραίτητες), ένα κατάλληλο string

που θα εμπεριέχει τα ονόματα των στηλών του πίνακα transportation και η δημιουργία ενός DefaultTableModel.

Στη συνέχεια διαμέσου ενός PreparedStatement, ενός executeQuery() και μιας while που απλά θα τρέχει εφόσον τα δεδομένα που θα βρίσκει «δεν είναι κενά» (το query θα ειναι select * from transportation;) , να παίρνουμε τις τιμές τύπου String που θα βρίσκει μέσα τον πίνακα με την συνάρτηση .getString(). Κατόπιν αυτά τα strings που ουσιαστικά είναι οι τιμές των στηλών του πίνακα θα θέτονται σε ένα μεγάλο string και θα γίνονται κάθε φορά add στο table model.

Τέλος όταν τρέξουμε το show transportation θα δούμε τον πίνακα των μεταφορών όπως ακριβώς υπάρχει και στο MySQL Workbench, σωστά τιτλοδοτημένος και οργανωμένος.

TRAVEL AGENT SIDE

Για τους ταξιδιωτικούς πράκτορες αρχικά δημιουργούμε το βασικό interface που οι βασικές του λειτουργίες είναι να δημιουργεί ενα destination, να εκτυπώνει τα ΔΙΚΑ του destinations και να δημιουργεί εισητήρια. Οπότε αρχικά φτιάχνουμε το κατάλληλο JFrame **(Εικόνα: 37**).

```
JFrame frame2 = new JFrame();
frame2.setSize(300, 450);
JPanel panel2 = new JPanel(new GridLayout(4, 1));
frame2.add(panel2, BorderLayout.CENTER);
JButton destination_create_a = new JButton("CREATE DESTINATION");
JButton print_destinations = new JButton("PRINT DESTINATIONS");
JButton create_ticket = new JButton("CREATE TICKET");
JButton exit = new JButton("EXIT");

//add everything to panel
panel2.add(destination_create_a);
panel2.add(print_destinations);
panel2.add(create_ticket);
panel2.add(exit);
```

Εικόνα 37: κεντρικο μενού – adminInterface

Η δημιουργία ενος destination έχει παρομοιο τρόπο με την εισαγωγή πχ customer , δηλαδή JFrame, JPanel και τα κατάλληλα components που συνθέτουν μια φόρμα. Όσο αφορά την σύνδεση και την εγγραφή ενος destination μπορούμε να δούμε οτι (**Εικόνα: 38**).

```
y {
  Class.forName("com.mysql.jdbc.Driver");
  try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "roo
      Statement stmt = con.createStatement();
      ResultSet rs = stmt.executeQuery("select * from destinations");
      String query = "INSERT INTO destinations (destination_name,arrival_destination,starting_point,destination_distance,time_start_
      PreparedStatement preparedStmt = con.prepareStatement(query);

      PreparedStatement pStmnt = null;
      String preQueryStatement = "select  * from destinations where destinations.destination_code = ?";
      pStmnt = con.prepareStatement(preQueryStatement);
      pStmnt.setString(1, destination_code);
      ResultSet rs1 = pStmnt.executeQuery();

      while (!rs1.next()) {

          preparedStmt.setString(1, destination_name);
          preparedStmt.setString(2, dest_arival);
          preparedStmt.setString(3, destination_start_point);
          preparedStmt.setString(4, destination_distance);
          preparedStmt.setString(5, time_start_h);
          preparedStmt.setString(6, time_start_minutes);
          preparedStmt.setString(7, time_end_hours);
          preparedStmt.setString(8, time_end_minutes);
          preparedStmt.setString(9, seat_cheapPrice);
          preparedStmt.setString(10, casualSeatsNumber);
          preparedStmt.setString(11, classA_seatsPrice);
          preparedStmt.setString(12, classA_seatsNumber);
          preparedStmt.setString(13, destination_code);
          preparedStmt.execute();
```

Εικόνα 38: εισαγωγη destination στην βαση δεδομενων μας

Η εκτύπωση destinations ειναι λιγο διαφορετική απο την απλή εισαγωγή εγγραφής στην βάση δεδομένων. Πιο συγκεκριμένα, δημιουργούμε ένα κατάλληλο JFrame, ενα κατάλληλο JTable. Ορίζουμε ενα string τα πεδία των στηλών που διαθέτει ο πίνακας που θέλουμε να εκτυπώσουμε, δηλαδή τα ονόματα των στηλών.

Στη συνέχεια με ενα query παίρνουμε όλα τα στοιχεία του πίνακα μας σε μορφές String χρησιμοποιώντας μια while για επανάληψη. Κάθε φορά που η while κάνει μια επανάληψη γράφει τα πάντα σε ένα πολύ μεγάλο string με την σειρά που τα βρίσκει στον πίνακα, τα εισάγει στη συνέχεια στο JTable και τα εκτυπώνει στην οθόνη «σαν να βλέπαμε την βάση δεδομένων μας απο το WorkBench».

```
try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
    JFrame frame_area = new JFrame();
    frame_area.setTitle("DESTINATIONS");
    frame_area.setSize(1100, 400);
    JTable jt = new JTable();
    JScrollPane jscrlp = new JScrollPane(jt, JScrollPane.VERTICAL_SCROLLBAR_ALWAYS, JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
    frame_area.add(jscrlp);
    String[] columnNames = {"ID", "NAME", " ARRIVAL", "S POINT", "D DISTANCE", "START T(H)", "START T(M)", "FINISH T(H)", "FINISH T(M)", "SEAT PRICE(C)", "SEAT PRICE
    DefaultTableModel tableModel = new DefaultTableModel(columnNames, 0);
    //get data from the table
    PreparedStatement statement = con.prepareStatement("SELECT * FROM destinations;");
    ResultSet result_statement = statement.executeQuery();

    while (result_statement.next()) {
        //get all values from the table
        String destination_id = result_statement.getString("destination_id");
        String destination_name = result_statement.getString("destination_name");
        String arrival_destination = result_statement.getString("arrival_destination");
        String starting_point = result_statement.getString("starting_point");
        String destination_distance = result_statement.getString("destination_distance");
        String time_start_h = result_statement.getString("time_start_h");
        String time_start_f = result_statement.getString("time_start_f");
        String time_end_h = result_statement.getString("time_end_h");
        String time_end_f = result_statement.getString("time_end_f");
        String seat_cheap_price = result_statement.getString("seat_cheap_price");
        String casual_seats_number = result_statement.getString("casual_seats_number");
        String classA_seat_price = result_statement.getString("classA_seatPrice");
        String classA_seatsNumber = result_statement.getString("classA_seatsNumber");
        String destination_code = result_statement.getString("destination_code");
        String[] data = {destination_id, destination_name, arrival_destination, starting_point, destination_distance, time_start_h, time_start_f, time_end_h, time_er
        tableModel.addRow(data);
    }
    jt.setModel(tableModel);
```

Εικόνα 39: κώδικας εκτύπωσης των destinations

Για παράδειγμα εαν θέλουμε να τα εκτυπώσουμε η μορφή με την οποία θα εμφανιστούν τα δεδομένα του πίνακα στην οθόνη φαίνονται παρακάτω πχ εάν συνδεθούμε με τον χρήστη kaliopi74 που διαθέτει κωδικό χρήστη 332211(**Εικόνα:40**).

DESTINATIONS

ID	NAME	ARRIVAL	S POINT	D DISTANCE	START T(H)	START T(M)	FINISH T(H)	FINISH T(M)	SEAT PRIC...	SEAT PRIC...	SEAT PRIC...	SEAT NUM...	CODE
1	Athens-Paris	22/10/19	Athens	300	12	30	14	30	65	25	160	4	10
2	Athens-Paris	21/10/19	Athens	300	12	30	14	30	65	25	160	4	11
3	Athens-Paris	26/10/19	Athens	300	12	30	14	30	65	25	160	4	12
4	Athens-Paris	19/10/19	Athens	300	12	30	14	30	65	25	160	4	13
5	Athens-Paris	12/10/19	Athens	300	12	30	14	30	65	25	160	4	14
6	Athens-Paris	11/10/19	Athens	300	12	30	14	30	65	25	160	4	15
7	Athens-Paris	29/10/19	Athens	300	12	30	14	30	65	25	160	4	16
8	Athens-Paris	30/10/19	Athens	300	12	30	14	30	65	25	160	4	17
9	Athens-Paris	31/10/19	Athens	300	12	30	14	30	65	25	160	4	18
10	Athens-Paris	7/10/19	Athens	300	12	30	14	30	65	25	160	4	19
11	Athens-Sw...	7/10/19	Athens	350	11	30	16	30	80	22	200	2	20
12	Athens-Sw...	12/10/19	Athens	350	11	30	16	30	80	22	200	2	21
13	Athens-Sw...	13/10/19	Athens	350	11	30	16	30	80	22	200	2	23
14	Athens-Sw...	19/10/19	Athens	350	11	30	16	30	80	22	200	2	24

Εικόνα 40: Η εκτύπωση των destinations

Για να γίνει η εκτύπωση πιο σωστή σε περίπτωση που έχουμε πολύ μεγάλο όγκο δεδομένων βάλαμε και JScrollBars με σκοπό να μπορούμε να κάνουμε κύλιση στον πίνακα.

Από την άλλη πλευρά η δημιουργία ticket γίνεται ακριβώς με τον ίδιο τρόπο με τον οποίο γίνεται οποιαδήποτε εγγραφή σε κάποιο πίνακα. Για παράδειγμα η εγγραφή στην βάση των tickets γίνεται εως εξής (**Εικόνα: 41**).

```
try {
    Class.forName("com.mysql.jdbc.Driver");
    try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("select * from tickets");
        String query = "INSERT INTO tickets (ticket_destination,ticket_airplaneCompany,ticket_agent_company,ticket_company_agent_id, ticket_price,ticket_c
        PreparedStatement preparedStmt = con.prepareStatement(query);

        PreparedStatement pStmnt = null;
        String preQueryStatement = "select  * from tickets where tickets.ticket_code = ?";
        pStmnt = con.prepareStatement(preQueryStatement);
        pStmnt.setString(1, ticket_code);
        ResultSet rs1 = pStmnt.executeQuery();

        while (!rs1.next()) {
            preparedStmt.setString(1, ticket_destination);
            preparedStmt.setString(2, ticket_airplaneCompany);
            preparedStmt.setString(3, ticket_agent_company);
            preparedStmt.setString(4, ticket_agent_id);
            preparedStmt.setString(5, ticket_price);
            preparedStmt.setString(6, ticket_code);
            preparedStmt.execute();
            System.out.println("TICKET CREATED SUCCESSFULLY");
            con.close();
        }
```

Εικόνα 41: η εγγραφή ticket στην βαση

Επιπρόσθετα, πάντα θυμόμαστε να κλείνουμε την βάση με την συνάρτηση .close() για να μην γίνει μπέρδεμα με τις παράλληλες εγγραφές, διαγραφές και τις λοιπές λειτουργίες.

CUSTOMER SIDE

Αρχικά έχουμε ένα βασικό JFrame στο οποίο διαφαίνονται οι λειτουργίες του πελάτη που ειναι show destinations, book a destination,show bookings, delete booking. Εδώ τα πράγματα είναι λιγάκι πιο περίπλοκα.

Όσο αφορά το book destination θα πρέπει αρχικά να επιλέξει ο customer ποιό destination θέλει και να τις αποθηκεύσουμε σε έναν πίνακα που ονομάζεται final_bookings. Αυτό γίνεται διότι πολύ πιθανόν ο πελάτης να θέλει να ακυρώσει μια κράτηση(booking) οπότε

1. Ο πελάτης μέσω κατάλληλης φόρμας δίνει τον κωδικό του destination που θέλει εκείνος

2. Βάσει του κωδικού αυτού(destination_code) γίνεται copy η γραμμή από το destination table στον πινακα final_bookings
3. Σβηνουμε την εγγραφή απο τον πίνακα destinations

Με αυτό τον τρόπο διατηρούμε τα δεδομένα της εγγραφής που «θέλουμε να σβήσουμε από τον πίνακα δρομολογίων – Ώστε εάν ένας νέος πελάτης εισέλθει στο σύστημα να μην μπορεί να πάρει το ίδιο destination» αλλά λόγω του οτι θα χρειαστούμε να μετρήσουμε τα χιλιομετρα που έκανε ένας πελάτης στην διάρκεια ταξιδίων του και άλλες παραμέτρους, πρέπει να αποθηκεύσουμε την εγγραφή αυτή στο final_bookings για παν ενδεχόμενο.

Επιπρόσθετα γράφεται στον πίνακα bookings κάποια βασικά στοιχεία του χρήστη που έκλεισε το destination. Άρα ενεπλέκονται συνολικά τρείς πίνακες

- destinations
- bookings
- final_bookings

Στην (**Εικονα:42**) βλέπουμε το copy της εγγραφής απο το destinations στον πίνακα final_bookings

```
if (rs.next()) {
    String destination_name = rs.getString("destination_name");
    String arrival_destination = rs.getString("arrival_destination");
    String starting_point = rs.getString("starting_point");
    String destination_distance = rs.getString("destination_distance");
    String time_start_h = rs.getString("time_start_h");
    String time_start_f = rs.getString("time_start_f");
    String time_end_h = rs.getString("time_end_h");
    String time_end_f = rs.getString("time_end_f");
    String seat_cheap_price = rs.getString("seat_cheap_price");
    String casual_seats_number = rs.getString("casual_seats_number");
    String classA_seat_price = rs.getString("classA_seatPrice");
    String classA_seatsNumber = rs.getString("classA_seatsNumber");
    String destination_code = rs.getString("destination_code");

    //SET THOSE VALUES TO final_bookings table
    String query = "INSERT INTO final_bookings (destination_name,arrival_destination,starting_point,destination_distance,time_sta
    PreparedStatement preparedStmt = con.prepareStatement(query);
    preparedStmt.setString(1, destination_name);
    preparedStmt.setString(2, arrival_destination);
    preparedStmt.setString(3, starting_point);
    preparedStmt.setString(4, destination_distance);
    preparedStmt.setString(5, time_start_h);
    preparedStmt.setString(6, time_start_f);
    preparedStmt.setString(7, time_end_h);
    preparedStmt.setString(8, time_end_f);
    preparedStmt.setString(9, seat_cheap_price);
    preparedStmt.setString(10, casual_seats_number);
    preparedStmt.setString(11, classA_seat_price);
    preparedStmt.setString(12, classA_seatsNumber);
```

Εικόνα 42:copy from destinations to final_bookings

Κατόπιν σβήνουμε την εγγραφή απο το destinations table ενώ γράφουμε κατάλληλη εγγραφή στον πίνακα bookings(**Εικόνα: 43**)

```
Class.forName("com.mysql.jdbc.Driver");
try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root"
    //delete the chosen booking from "destinations" table
    PreparedStatement pStmnt = null;
    String preQueryStatement = "delete  from destinations where destination_code = ?";
    pStmnt = con.prepareStatement(preQueryStatement);
    pStmnt.setString(1, booking_destination_code);
    pStmnt.executeUpdate();
    //insert the booking to table "bookings"
    String query = "INSERT INTO bookings (booking_destination,customer_username,destination_code)" + "VALUES (?,?,?)";
    PreparedStatement preparedStmt = con.prepareStatement(query);
    PreparedStatement pStmnt1 = null;
    String preQueryStatement1 = "select  * from bookings where bookings.destination_code = ?";
    pStmnt1 = con.prepareStatement(preQueryStatement1);
    pStmnt1.setString(1, booking_destination_code);
    ResultSet rs1 = pStmnt1.executeQuery();
    //add row to "bookings" table
    while (!rs1.next()) {
        //checking if another booking has the same destination code [ if not, then we add the record to table "bookings"]
        preparedStmt.setString(1, booking_destination);
        preparedStmt.setString(2, booking_customer_username);
        preparedStmt.setString(3, booking_destination_code);
        preparedStmt.execute();
        System.out.println("BOOKING CREATED SUCCESSFULLY");
        con.close();
    }
}
```

Εικόνα 43: διαγραφή απο destinations & εισαγωγη εγγραφης στο bookings

Όσο αφορά το show bookings ειναι απλά μια εκτύπωση όπως τις άλλες φορές με κατάλληλη μορφοποίηση του JTable. Φυσικά από την αρχή της εφαρμογής περάσαμε το username του customer μεσα στον actionListener() για να μπορούμε εδώ να κοιτάμε μόνο τις κρατήσεις(bookings) του πελάτη που έχει συνδεθεί στη βάση δεδομένων και όχι αλλων πελατών **(Εικόνα: 44)**

```
//checking user's booking query in order to show ONLY THE BOOKINGS OF THIS SPECIFIC CUSTOMER WHICH HAS BEEN LOGGED IN
PreparedStatement pStmnt = null;
String preQueryStatement = "select  * from bookings where bookings.customer_username = ?";
pStmnt = con.prepareStatement(preQueryStatement);
pStmnt.setString(1, username_s);
ResultSet rs1 = pStmnt.executeQuery();

String[] columnNames = {"ID", "BOOKING DEST", " USERNAME", "DEST CODE"};
DefaultTableModel tableModel = new DefaultTableModel(columnNames, 0);
//get data from the table
PreparedStatement statement = con.prepareStatement("SELECT * FROM bookings;");
ResultSet result_statement = statement.executeQuery();

while (rs1.next()) {
    //get all values from the table
    String booking_id = rs1.getString("booking_id");
    String booking_destination = rs1.getString("booking_destination");
    String customer_username = rs1.getString("customer_username");
    String destination_code = rs1.getString("destination_code");
    statement.execute();
    String[] data = {booking_id, booking_destination, customer_username, destination_code};
    tableModel.addRow(data);

}
```

Εικόνα 44: show bookings

Όσο αφορά την διαγραφή της κράτησης(booking) του πελάτη χρησιμοποιούμε απλά ένα query delete/from κατάλληλο destination code από τον πίνακα final_bookings(**Εικόνα: 45**)

```
//add actionListener
delete_booking_b.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent ae) {
        String delete_dest_code = delete_booking_f.getText();
        try {
            Class.forName("com.mysql.jdbc.Driver");
            try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb?autoReconnect=true&useSSL=false", "root", "root")) {
                //delete the FINAL booking from the final_bookings table
                PreparedStatement pStmnt = null;
                String preQueryStatement = "delete  from final_bookings where final_bookings.destination_code = ?";
                pStmnt = con.prepareStatement(preQueryStatement);
                pStmnt.setString(1, delete_dest_code);
                pStmnt.executeUpdate();
                System.out.println("BOOKING DELETED SUCCESSFULLY");
```

Εικόνα 45:delete booking

ΣΥΜΠΕΡΑΣΜΑΤΑ

Στην σημερινή εποχή παρατηρούμε οτι έχει αυξηθεί ο όγκος δεδομένων που χρησιμοποιούμε καθημερινά οπότε είναι πολύ δύσκολο να χειρισθεί εύκολα. Επομένως είναι απαραίτητο να υπάρχουν συστήματα κατάλληλα που να μπορούν να ομαδοποιούν τα δεδομένα κατάλληλα.

Επομένως, με την χρήση των βάσεων δεδομένων η ζωή μας έγινε ευκολότερη και συνάμα με την δύναμη του προγραμματισμού μπορούμε να επέμβουμε πάνω στα δεδομένα όπως εμείς το θέλουμε καθε φορα. Η χρησιμότητα των βάσεων δεδομένων ειναι ευρεία και βρίσκεται σχεδόν σε όλες τις εφαρμογές που απαιτούν να γίνουν κατάλληλες πράξεις στα δεδομένα ή απλα για αποθήκευση στοιχείων χρηστών, πραγμάτων κ.α σε ένα σύστημα.

Η εφαρμογή θα μπορούσε φυσικά να γίνει και με πολύ διαφορετικό τρόπο ή και ακόμη με άλλη γλώσσα προγραμματισμού ή ακόμη και με διαφορετικό τρόπο αποθήκευσης δεδομένων όπως είναι για παράδειγμα το XML

ΒΙΒΛΙΟΓΡΑΦΙΑ

- Bachman., C. W. (1973). The Programmer as Navigator. Communications of the ACM. 16 (11). pp. 653–658.
- Beaumont., R.A.(1942). Aeronautical Engineering
- Becker, B. (1967). Dreams and Realities of the Conquest of the Skies. New York: Atheneum. pp. 124-125
- Benedetti.,F. (2003). FAI NEWS: 100 Years Ago, the Dream of Icarus Became Reality
- Beynon-Davies, P. (2003). Database Systems (3rd Ed.). Palgrave Macmillan century-of-flight. Aviation timeline 1910 [Accessed 30 October 2019 from here]
- Chapple., M. (2005). SQL Fundamentals
- Childs., D. L. (1968a). Description of a set-theoretic data structure"
- Chong., R.. F.; Wang, X., Dang, M., Snow., D. R. (2007). Introduction to DB2
- Connolly., T. M., Begg., C. E. (2014). Database Systems – A Practical Approach to Design Implementation and Management (6th ed.). Pearson
- Crouch, T. (1982). Bleriot XI, The Story of a Classic Aircraft. Smithsonian Institution Press. pp. 22-22
- Date., C. J. (2003). An Introduction to Database Systems (8th ed.). Pearson.
- Davies., A. (2010). Battle of the Jumbo Jets: Airbus A380 Vs. Boeing 747-8I
- Dwyer, L.(1968). Sir George Cayley: The Father of Aviation. The Journal of San Diego History. 14(3).
- Encyclopedia Britannica. Sir George Cayley
- Federal Aviation Administration. Aviation Maintenance Technician Handbook - Airframe

- Gibbs., S., Charles. H. (1959). Hops and Flights: A roll call of early powered take-offs
- Grewe., V., Brunner., M,. Dameris., J. L., Grenfell., R., Hein., D., Shindell., J., Staehelin., D. (2001). Origin and variability of upper tropospheric nitrogen oxides and ozone at northern mid-latitudes. Atmospheric Environment. 35 (20). pp. 3421–33
- Inglis, A. (1983). Hargrave, Lawrence (1850–1915). Australian Dictionary of Biography. Melbourne University Press
- Joel, C.(2015). Warning about SSL connection when connecting to MySQL Database. StackOverFlow [Accessed 10 October 2019 from here]
- Paur., J. (2010).1919: First Nonstop Flight Crosses Atlantic
- Penner., J. E., Lister., D., Griggs., David., J., Dokken., D. J., McFarland., M. (1999). Aviation and the Global Atmosphere
- Spaight., J.(1914). Aircraft In War. London: MacMilian and Co. pp. 3
- Technology Museum of Thessaloniki. Archytas of Tarentum, Technology Museum of Thessaloniki, Macedonia, Greece.
- White., L. (1961). Eilmer of Malmesbury, an Eleventh Century Aviator: A Case Study of Technological Innovation, Its Context and Tradition. Technology and Culture. 2 (2). pp. 97–111