**Αποτελέσματα των παιγνίων**

**της ΟΠΑΠ Α.Ε.**

**ΕΑΠ – ΠΛΗ 24**

**3η ΓΡΑΠΤΗ ΕΡΓΑΣΙΑ 2021-2022**

|  |  |  |  |
| --- | --- | --- | --- |
| **Νο** | **Ονοματεπώνυμο** | | **Α.Μ.** |
| 1 | ΕΜΜΑΝΟΥΗΛ ΠΙΤΥΑΝΟΣ | | 138573 |
| 2 | ΕΥΑΓΓΕΛΟΣ ΜΠΑΤΣΑΛΗΣ | | 119181 |
| 3 | ΚΩΝΣΤΑΝΤΙΝΟΣ ΒΡΕΖΑΣ | | 101855 |
| **ΣΧΟΛΙΑ ΠΡΟΣ ΤΟΝ ΚΑΘΗΓΗΤΗ** | | Το αρχείο «Οδηγίες για έξτρα βιβλιοθήκες και σύνδεση με την ΒΔ.docx», δημιουργήθηκε με βάση το αντίστοιχο αρχείο της περσυνής 2ης καλύτερης εργασίας (PLH24\_3GE\_VASILEIADOY). | |

***Υπεύθυνη Δήλωση Φοιτητή****: Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία αυτής της εργασίας, είναι πλήρως αναγνωρισμένη και αναφέρεται, είτε στο σημείο «Σχόλια προς καθηγητή», είτε μέσα στην εργασία. Επίσης, έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς, είτε παραφρασμένες. Επίσης, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη Θεματική Ενότητα.*

🗹 Συμφωνώ και αποδέχομαι την ανωτέρω δήλωση

Δε συμφωνώ και δεν αποδέχομαι την ανωτέρω δήλωση (στην περίπτωση αυτή, ο Κ-Σ έχει δικαίωμα να μην αξιολογήσει την εργασία του φοιτητή)

|  |
| --- |
| Ημερομηνία ανακοίνωσης εργασίας:, 24/01/2022  Ημερομηνία παράδοσης εργασίας: 09/03/2022  ΔΕΝ ΘΑ ΔOΘΟΥΝ ΠΑΡΑΤΑΣΕΙΣ |

ΣΤΟΙΧΕΙΑ ΠΟΥ ΣΥΜΠΛΗΡΩΝΕΙ Ο ΚΑΘΗΓΗΤΗΣ

|  |  |
| --- | --- |
| **ΗΜΕΡΟΜΗΝΙΑ ΑΞΙΟΛΟΓΗΣΗΣ** |  |
| **ΒΑΘΜΟΣ** |  |

**ΣΧΟΛΙΑ ΠΡΟΣ ΦΟΙΤΗΤΗ / ΦΟΙΤΗΤΡΙΑ**

##### Αναλυτική Αξιολόγηση

|  |  |  |  |
| --- | --- | --- | --- |
| **Άσκηση** | **Περιγραφή** | **Ποσοστό** | **Βαθμός** |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 1 | Διαχείριση του Έργου με έμφαση στη χρήση του εργαλείου συνεργασίας | 25 |  |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2.Α | Διάγραμμα Κλάσεων και Υλοποίηση Κλάσεων σε Java (περιλαμβάνει χρήση API και επεξεργασία json) | 20 |  |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2.Β | Δημιουργία GUI Εφαρμογής | 15 |  |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2.Γ | Παρουσίαση Στατιστικών Στοιχείων και Γραφημάτων | 20 |  |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2.Δ | Συνολικός Έλεγχος και Εκτέλεση της Εφαρμογής | 15 |  |
| ΔΡΑΣΤΗΡΙΟΤΗΤΑ 3 | Κριτικός ΑπολογισμόςτουΈργου | 5 |  |
|  | **Σύνολο** | 100 |  |

ΠΕΡΙΕΧΟΜΕΝΑ

[**1**](#_heading=h.gjdgxs) **ΕΙΣΑΓΩΓΗ 4**

[**2**](#_heading=h.30j0zll) **ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΟΥ 5**

[2.1](#_heading=h.1fob9te) Υπολογισμός της απαιτούμενης προσπάθειας ανά απαίτηση 5

[2.2](#_heading=h.3znysh7) Υπολογισμός των προτεραιοτήτων 5

[2.3](#_heading=h.2et92p0) Χρονοδιάγραμμα του έργου 5

[2.4](#_heading=h.tyjcwt) To product backlog 5

[2.5](#_heading=h.3dy6vkm) Οργάνωση ομάδος και αναθέσεις αρμοδιοτήτων 5

[2.6](#_heading=h.1t3h5sf) Παρακολούθηση της προσπάθειας κατά τη διάρκεια του έργου 5

[2.7](#_heading=h.4d34og8) Χρήση εργαλείου trello 6

[**3**](#_heading=h.2s8eyo1) **ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2 - ΥΛΟΠΟΙΗΣΗ ΕΡΓΟΥ 7**

[*3.1*](#_heading=h.17dp8vu) Ερώτημα Α– Διάγραμμα Κλάσεων και Υλοποίηση Κλάσεων σε Java 7

[3.2](#_heading=h.3rdcrjn) Ερώτημα Β – Δημιουργία GUI Εφαρμογής 7

[3.3](#_heading=h.26in1rg) Ερώτημα Γ – Παρουσίαση Στατιστικών Στοιχείων και Γραφημάτων 7

[3.4](#_heading=h.lnxbz9) Ερώτημα Δ – Συνολικός Έλεγχος και Εκτέλεση της Εφαρμογής 7

[**4**](#_heading=h.35nkun2) **ΚΡΙΤΙΚΟΣ ΑΠΟΛΟΓΙΣΜΟΣ ΤΟΥ ΕΡΓΟΥ 8**

[**5**](#_heading=h.1ksv4uv) **ΑΝΑΦΟΡΕΣ 9**

# ΕΙΣΑΓΩΓΗ

*[Δώστε μια σύντομη περιγραφή του συστήματος. Παρουσιάστε τους βασικούς στόχους του έργου, καθώς και τις παραδοχές σας. Αναφέρετε τα χαρακτηριστικά και τις λειτουργίες που πρέπει να επιτελεί το έργο ώστε να ανταποκρίνεται στις απαιτήσεις και τις ανάγκες του πελάτη. Αναφέρετε με ποιόν/ποιους τρόπο/τρόπους το έργο ικανοποιεί αυτές τις απαιτήσεις.]*

Το σύστημα που υλοποιήθηκε αποτελεί μία γραφική διεπαφή χρήστη με σκοπό την εύκολη και γρήγορη εύρεση, ανάγνωση και αποθήκευση των αποτελεσμάτων του παιχνιδιού ΤΖΟΚΕΡ του οργανισμού ΟΠΑΠ καθώς και την προβολή στατιστικών στοιχείων και έξοδο αυτών σε μορφή pdf.

Βασικοί στόχοι του έργου είναι να καλυφτούν οι προδιαγραφές που αφορούν την λειτουργικότητα του συστήματος οι οποίες είναι:

1. Διαχείριση δεδομένων ΤΖΟΚΕΡ.

1α.Εμφάνιση οθόνης και δυνατότητα επιλογής από τον χρήστη των δεδομένων που θα αντληθούν είτε εισάγοντας το Idενός παιχνιδιού είτε ένα εύρος ημερομηνιών.

1β. Τα δεδομένα που θα αντληθούν είναι οι τυχεροί αριθμοί, βασικές κατηγορίες επιτυχιών όπου για κάθε κατηγορία θέλουμε τον αριθμό των επιτυχιών και τα κέρδη ανά επιτυχία.

1γ. Δυνατότητα αποθήκευσης των δεδομένων που ανακτήθηκαν σε βάση δεδομένων.

1δ. Δυνατότητα διαγραφής ενός παιχνιδιού ή παιχνιδιών εντός εύρους ημερομηνιών.

2.Προβολή δεδομένων ΤΖΟΚΕΡ ανά μήνα για συγκεκριμένο έτος.

2α. Δυνατότητα επιλογής από μία λίστα το έτος και στη συνέχεια του μήνα που επιθυμεί ο χρήστης για προβολή συγκεντρωτικών δεδομένων.

2β. Τα συγκεντρωτικά δεδομένα αφορούν στο πόσα παιχνίδια έγιναν, πόσα χρήματα διανεμήθηκαν και πόσα ΤΖΑΚ-ΠΟΤ έγιναν.

1. Προβολή στατιστικών δεδομένων ΤΖΟΚΕΡ και εκτύπωση σε αρχείο pdf.

3α. Προβολή σε μορφή πίνακα, των διαθέσιμων από το apiτου ΟΠΑΠ στατιστικών, των αριθμών που έχουν κληρωθεί και δυνατότητα εκτύπωσης του παραγόμενου πίνακα σε αρχείο pdf.

3β. Δυνατότητα προβολής στατιστικών στοιχείων, από τα δεδομένα που είναι στη βάση δεδομένων, σε γραφική μορφή.

3γ. Δυνατότητα επιλογής του χρήστη ποια στατιστικά στοιχεία και για ποιο χρονικό διάστημα θα παραχθούν.

3δ. Τα στατιστικά στοιχεία αφορούν τους πέντε ποιο συχνά εμφανιζόμενους αριθμούς, τους πέντε ποιο συχνά εμφανιζόμενους αριθμούς jokerκαι το μέσο όρο κερδών ανά κατηγορία.

1. Δυνατότητα εξόδου του χρήστη και κλείσιμο της εφαρμογής από την αρχική οθόνη.

Οι παραπάνω απαιτήσεις αποτελούν τις ελάχιστες απαιτήσεις για να θεωρηθεί η εφαρμογή αποδεχτή από τον πελάτη και για το λόγο αυτό το έργο περιλαμβάνει από την αρχική του σχεδίαση μέχρι την υλοποίηση όλες αυτές τις απαιτήσεις.

# ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΟΥ

## Υπολογισμός της απαιτούμενης προσπάθειας ανά απαίτηση

*[Θα πρέπει να υπολογίσετε την απαιτούμενη προσπάθεια για το έργο σε ανθρωποημέρεςμε τη χρήση της μεθόδου planningpoker. Να τεκμηριώστε τις παραδοχές σας, το υλικό στο οποίο βασιστήκατε για την εφαρμογή της μεθόδου, καθώς και τον τρόπο εργασίας. Να παραθέσετε σχετικές αναφορές. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*

Η εκτίμηση της προσπάθειας που απαιτείται για το έργο σε ανθρωποημέρες έγινε με την χρήση του planningpoker <https://www.planningpoker.com> .

Έπειτα από επικοινωνία μέσω τηλεδιάσκεψης ορίστηκε η ώρα για την διεξαγωγή του

Παιχνιδιού με κάρτες T-shirt

Το αποτέλεσμα του πρώτου γύρου είχε ως εξής:

|  |  |  |  |
| --- | --- | --- | --- |
| Απαιτήσεις | Scrum master  ΕΜΜΑΝΟΥΗΛ ΠΙΤΥΑΝΟΣ | 1ο Μέλος  ΕΥΥΑΓΓΕΛΟΣ ΜΠΑΤΣΑΛΗΣ | 2ο Μέλος  ΒΡΕΖΑΣ ΚΩΝΣΤΑΝΤΙΝΟΣ |
| R1 | XS | S | M |
| R2 | XL | XL | XXL |
| R3 | L | XL | XXL |
| R4 | L | L | XL |
| R5 | XL | XXL | XXL |

Το αποτέλεσμα του δεύτερου γύρου και τελικού αποτελέσματος είχε ως εξής:

|  |  |  |  |
| --- | --- | --- | --- |
| Απαιτήσεις | Scrum master  ΕΜΜΑΝΟΥΗΛ ΠΙΤΥΑΝΟΣ | 1ο Μέλος  ΕΥΥΑΓΓΕΛΟΣ ΜΠΑΤΣΑΛΗΣ | 2ο Μέλος  ΒΡΕΖΑΣ ΚΩΝΣΤΑΝΤΙΝΟΣ |
| R1 | XS | XS | XS |
| R2 | XL | XL | XL |
| R3 | XL | XL | XL |
| R4 | L | L | L |
| R5 | XXL | XXL | XXL |

Οι παραδοχές που έγιναν από την ομάδα ήταν ότι οκτώώρες ενασχόλησης θα λογίζονται ως χρόνος μίας ανθρωποημέρας.

Επικοινωνία δύο φορές την εβδομάδα.

Προσπάθεια τήρησης από κάθε μέλος της ημερήσιας ενασχόλησης με το έργο.

## Υπολογισμός των προτεραιοτήτων

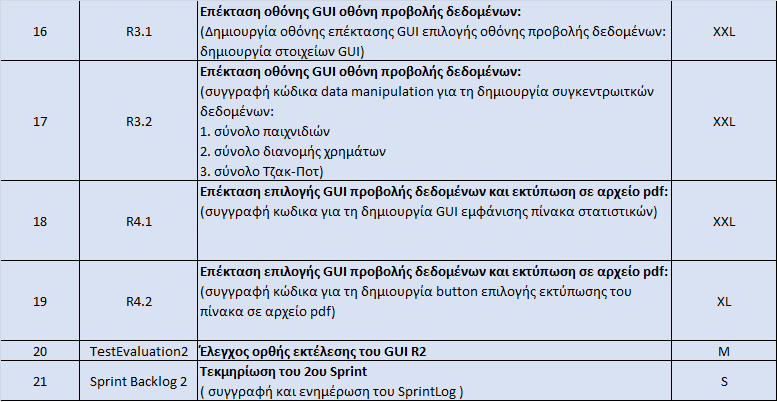
*[Θα πρέπει να υπολογίσετε την προτεραιότητα της κάθε απαίτησης χρησιμοποιώντας τη μέθοδο PriorityPoker. Στο σημείο αυτό θα παρουσιάσετε το τελικό αποτέλεσμα από την εφαρμογή της μεθόδου (ποιες απαιτήσεις θα υλοποιηθούν σε κάθε επανάληψη), καθώς και παρατηρήσεις σχετικές με τον τρόπο εργασίας σας. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*

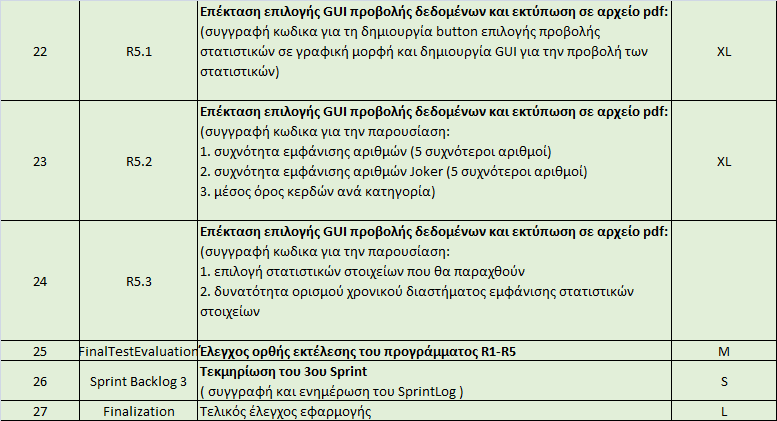
Χρησιμοποιώντας για ακόμα μία φορά planningpoker αυτή τη φορά για τον υπολογισμό προτεραιότητας των απαιτήσεων το τελικό αποτέλεσμα φαίνεται στην παρακάτω εικόνα.

Για το Sprint 1:



Για το Sprint 2:

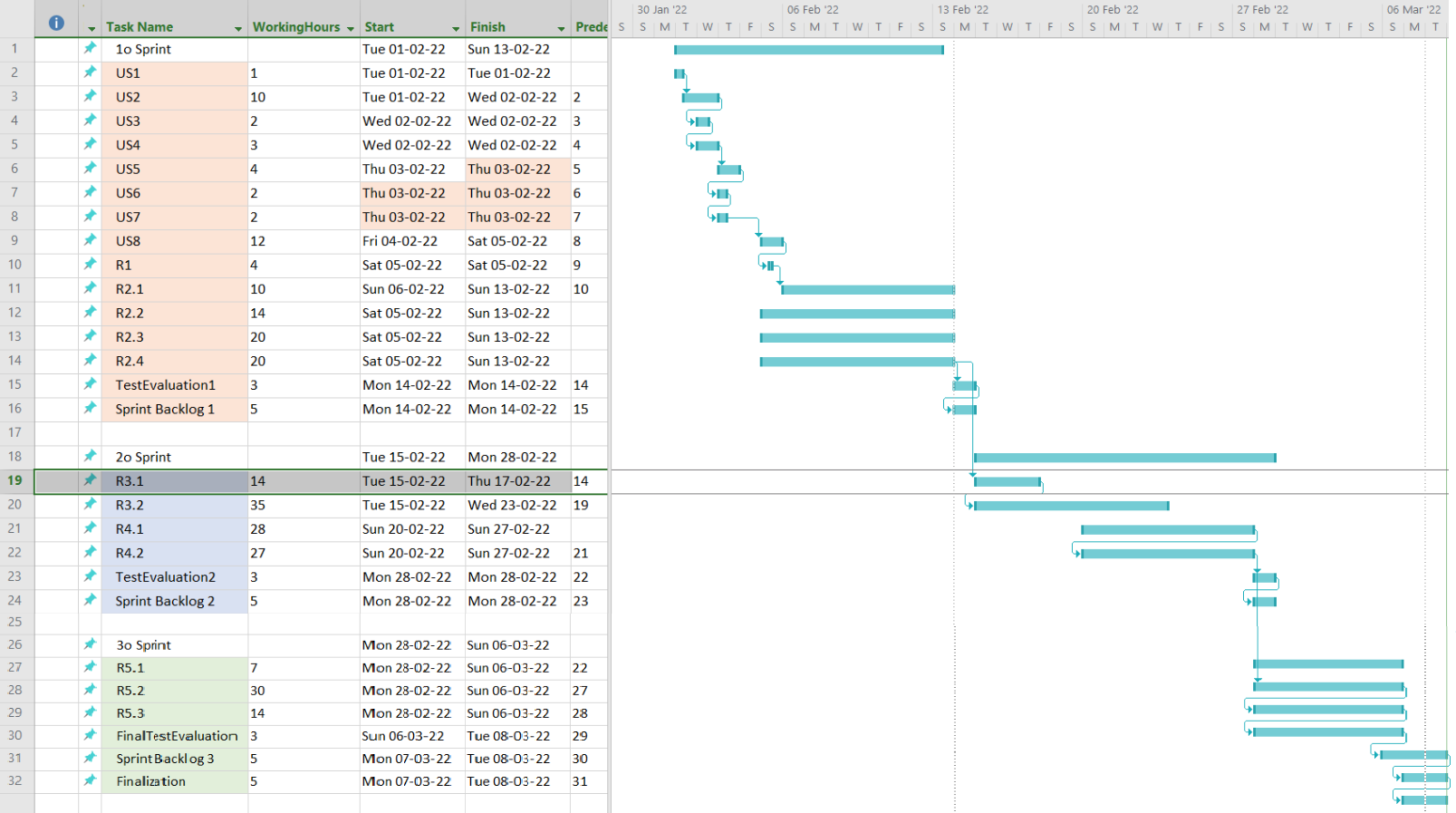
Για το Sprint 3:



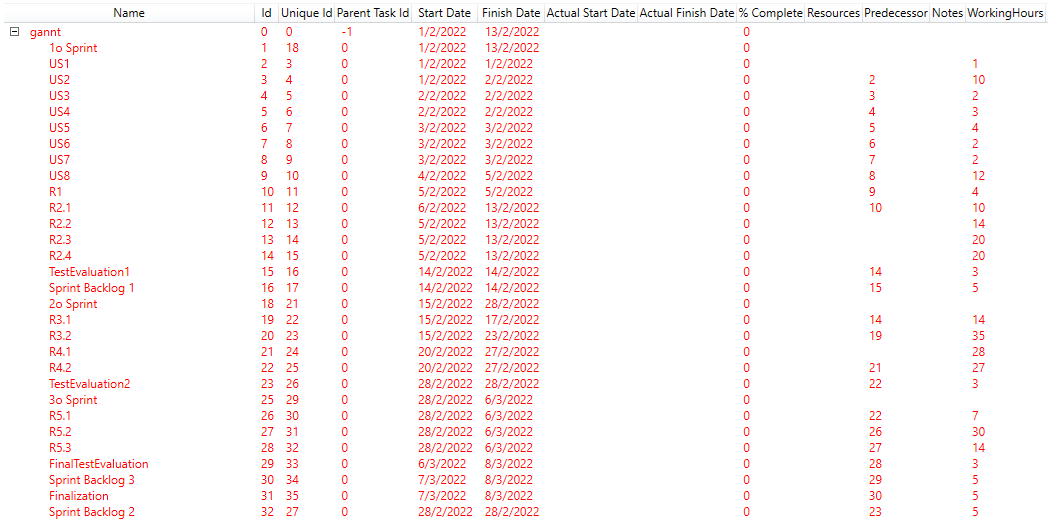
## Χρονοδιάγραμμα του έργου

*[Εισάγετε σχετική εικόνα. Στην Εικόνα θα πρέπει να φαίνεται το χρονοδιάγραμμα του έργου σε μορφή Gantt (που θα έχει κατασκευαστεί με χρήση του σχετικού εργαλείου). Παρουσιάστε με συντομία και με μορφή κειμένου τις βασικές δραστηριότητες της κάθε επανάληψης. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*

Το διάγραμμα Ganttέχει ως εξής:



Τα περιεχόμενα του διαγράμματος Ganttφαίνονται στον παρακάτω πίνακα.

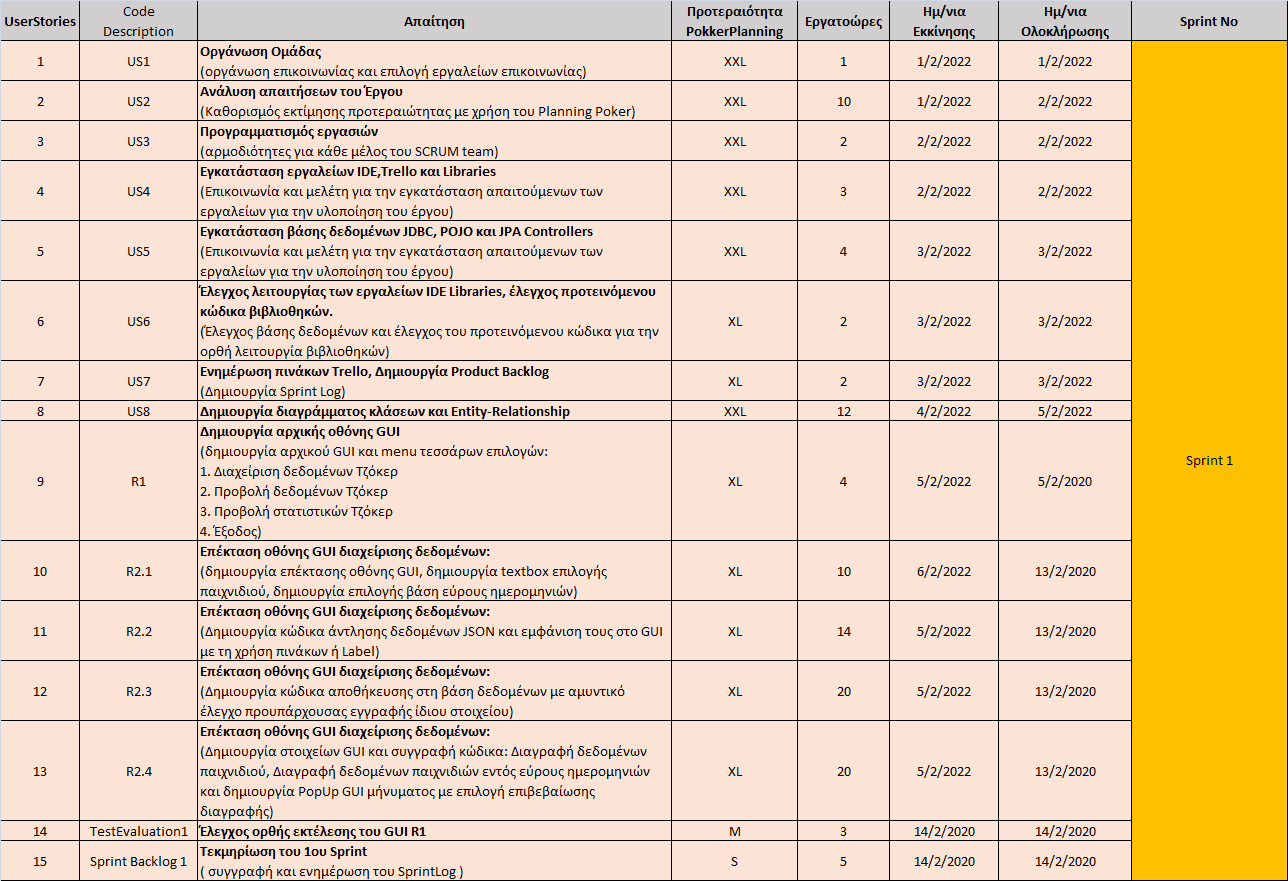


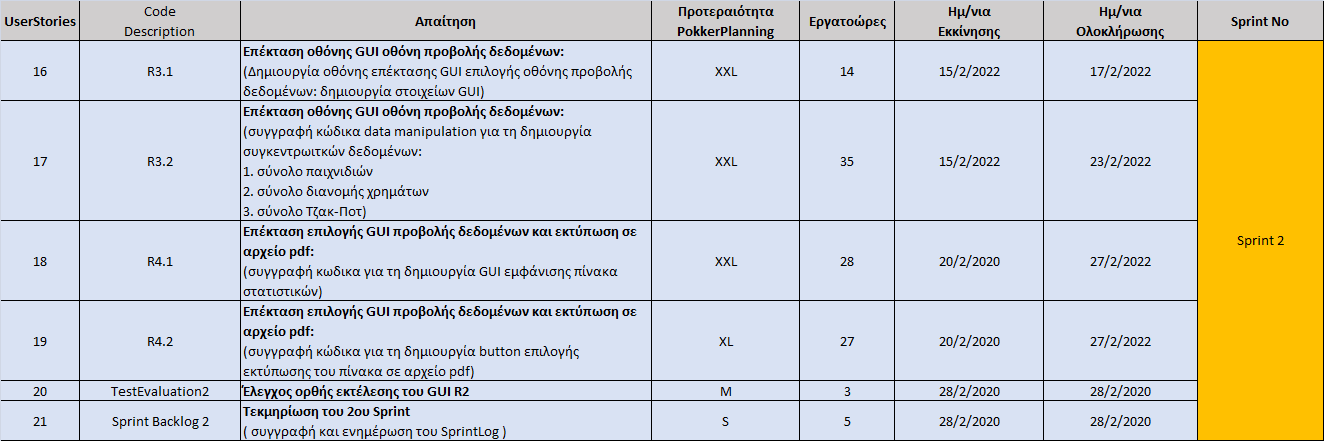
Όπως φαίνεται και στον πίνακα το έργο ολοκληρώθηκε σε τρία Sprint

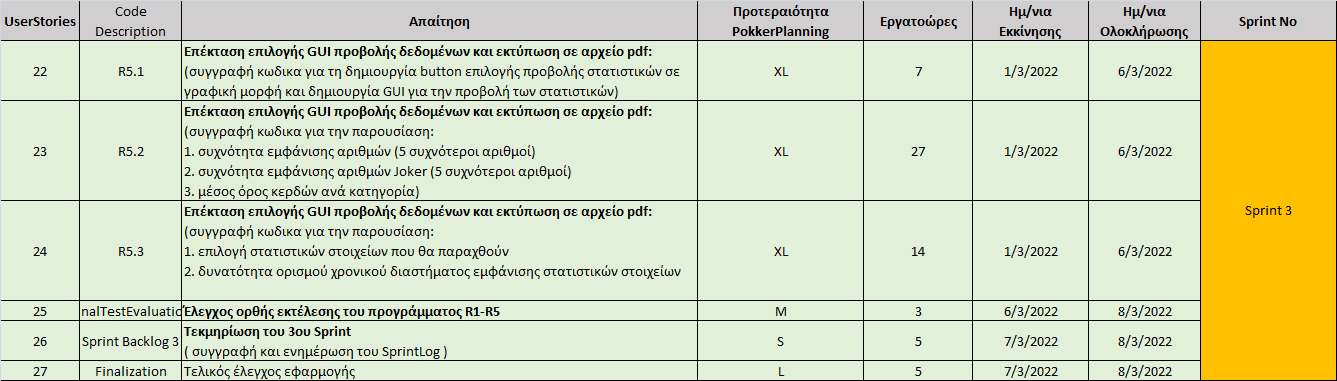
## To product backlog

*[Θα πρέπει να παρουσιάσετε το productbacklogσε μορφή φύλλου εργασίας (xls) που θα περιέχει τρία φύλλα εργασίας, ένα για κάθε επανάληψη, μαζί με τις συμπληρωματικές/βοηθητικές πληροφορίες που αποφασίσατε να συλλέξετε. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*

Το αρχείο product backlog είναι σε μορφή (xls) μέσα στο φάκελο του πρότζεκτ με ονομασία product backlog.xls, παρακάτω παραθέτουμε εικόνες από το αρχείο.







## Οργάνωση ομάδος και αναθέσεις αρμοδιοτήτων

*[Αναφέρετε την οργάνωση της ομάδας του έργου, καθώς και τους ρόλους των μελών της ομάδας. Πιο συγκεκριμένα:*

* *Ποιος ήταν ο ρόλος του κάθε μέλους της ομάδας;*
* *Ποιος ήταν ο υπεύθυνος για την υλοποίηση της κάθε απαίτησης;*
* *Αναφέρετε τον τρόπο επικοινωνίας των μελών της ομάδας, τη διαχείριση διαφορών μεταξύ των μελών, κ.λπ.και ποιες από τις πρακτικές του SCRUMχρησιμοποιήσατε;*

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*

## Οι ρόλοι κάθε μέλους ήταν :

ScrumMaster: Πιτυανός Εμμανουήλ έμπειρος στον προγραμματισμό καθοδήγησε και βοήθησε τα υπόλοιπα μέλη στο να επιτύχουν την υλοποίηση του κώδικα που είχαν αναλάβει. Ταυτόχρονα ασχολήθηκε ως προγραμματιστής, αναλυτής συστήματος, σχεδιαστής και δοκιμαστής.

Ευάγγελος Μπατσαλής ασχολήθηκε προγραμματιστής, αναλυτής συστήματος, σχεδιαστής και δοκιμαστής.

Βρέζας Κωνσταντίνος ασχολήθηκε ως προγραμματιστής και δοκιμαστής.

Για την R1 απαίτηση ασχολήθηκε ο Πιτυανός Εμμανουήλ.

Για την R2 απαίτηση ασχολήθηκε ο Πιτυανός Εμμανουήλ, Ευάγγελος Μπατσαλής, Βρέζας Κωνσταντίνος.

Για την R3 απαίτηση ασχολήθηκε ο Πιτυανός Εμμανουήλ.

Για την R4 απαίτηση ασχολήθηκε ο Ευάγγελος Μπατσαλής.

Για την R5 απαίτηση ασχολήθηκε ο Πιτυανός Εμμανουήλ,Ευάγγελος Μπατσαλής, Βρέζας Κωνσταντίνος.

Η επικοινωνία της ομάδας δύο φορές την εβδομάδα οργανώνονταν με emailκαι εκτελούνταν μέσω τηλεδιάσκεψης. Στις τηλεδιασκέψεις να ορίσουμε με ποιο κομμάτι του κώδικα θα ασχοληθεί το κάθε μέλος και να επιλύσουμε κάποιες απορίες.

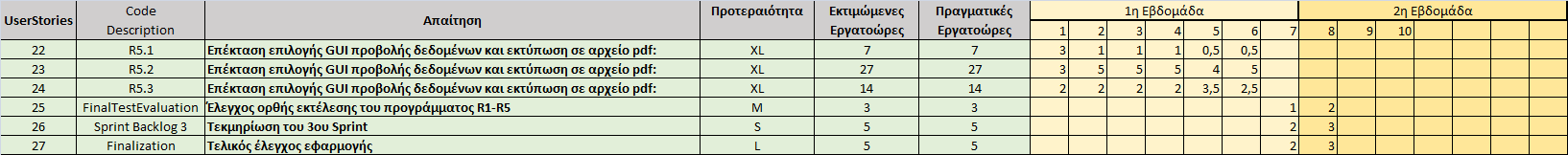
Χρησιμοποιήσαμε το PlanningPoker για τον καθορισμό των προτεραιοτήτων.

## Παρακολούθηση της προσπάθειας κατά τη διάρκεια του έργου

*[Παρουσιάστε το χρόνο υλοποίησης ανά απαίτηση/ανά επανάληψη (sprint) σε σύγκριση με αυτά που είχαν προϋπολογιστεί, το διάγραμμα κατανάλωσης προσπάθειας (burntdownchart) ανά sprint, καθώς και την ταχύτητα υλοποίησης (velocity). Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*







Η ταχύτητα υλοποίησης είναι:

Για το 1o Sprint

1+10+2+3+4+2+2+12+4+10+14+20+20+3+5=112.

Για το 2o Sprint

14+35+28+27+3+5=112.

Για το 3o Sprint

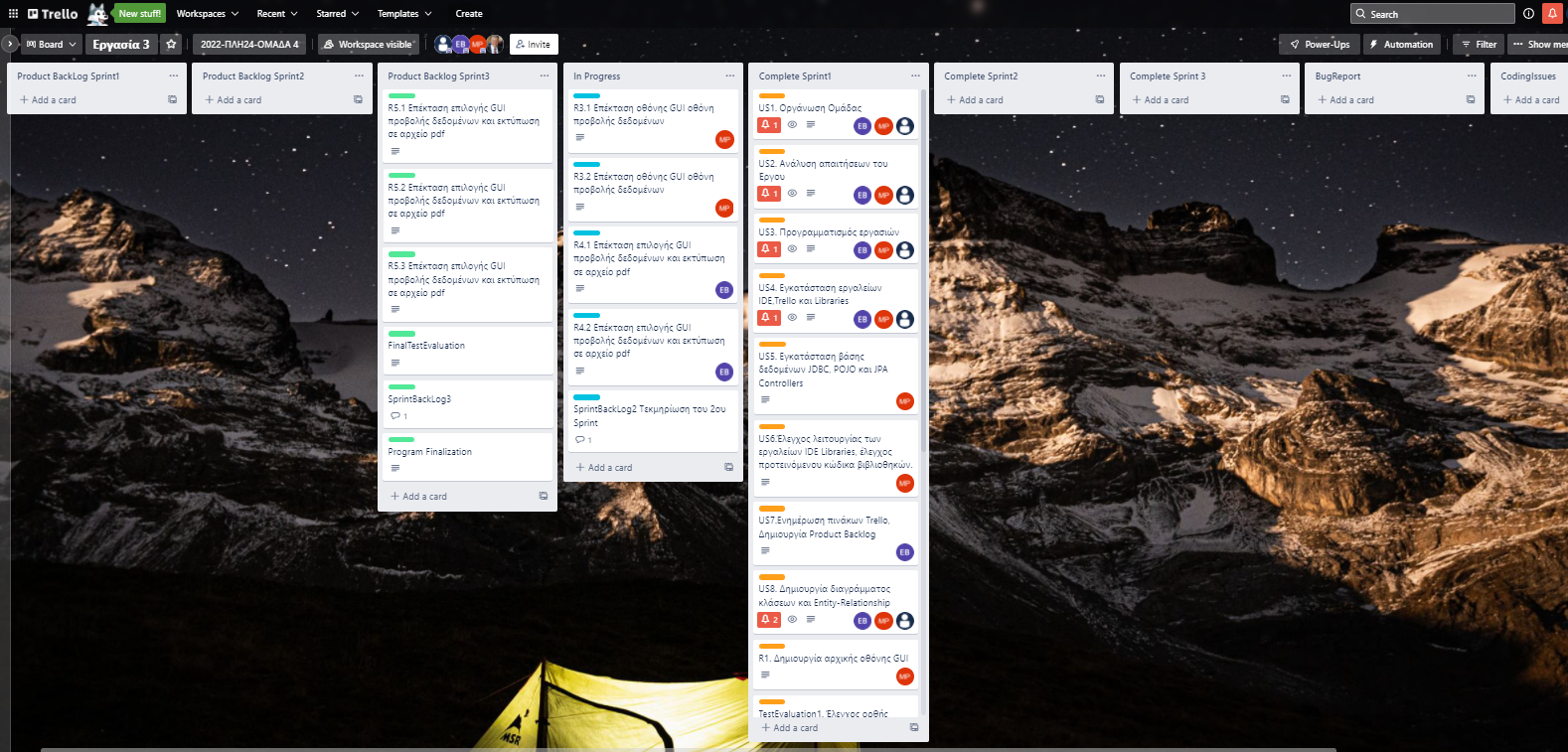
7+27+14+3+5+5=61.

## Χρήση εργαλείου trello

Παρουσιάστε τον τρόπο χρήσης του εργαλείου:

1. Πώς οργανώσατε το AgileBoard, ποιες λίστες χρησιμοποιήσατε. Δώστε ενδεικτικό Screenshot.
2. Πώς ορίσατε τις κατηγορίες δραστηριοτήτων
3. Πώς καταγράψατε τον χρόνο που χρειάστηκε η κάθε δραστηριότητα
4. Πώς συνεργάστηκε η ομάδα στο AgileBoard
5. Πως εκμεταλλευτήκατε τις λειτουργίες του Trello. Ποιες σας φάνηκες χρήσιμες, ποιες εύκολες στη χρήση και ποιες όχι.
6. Αναλύστε ποιες συνδέσεις με τρίτες εφαρμογές χρησιμοποιήσατε και πώς. Πχ με το github.

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ]*



1. Όπως φαίνεται και στο Screenshotχρησιμοποιήσαμε συνολικά εννέα λίστες, τρεις λίστες μία για κάθε Sprint, μία λίστα για τις δραστηριότητες που είναι σε εξέλιξη, τρεις λίστες μία για κάθε Sprintπου έχει ολοκληρωθεί, μία λίστα για αναφορές προβλημάτων κατά την εκτέλεση της εφαρμογής και μία λίστα για αναφορά προβλημάτων κωδικοποίησης.

2. Ορίσαμε τις δραστηριότητες σύμφωνα με την αναγκαία σειρά υλοποίησης αυτών και με εκτίμηση του χρόνου περάτωσης της κάθε δραστηριότητας.

3. Κάθε μέλος κρατούσε ημερολόγιο με τις ώρες που χρειάστηκε να δουλέψει σε κάθε δραστηριότητα.

4 Η ομάδα συνεργάστηκε στο AgileBoard για τον διαχωρισμό των απαιτήσεων και τον ορισμό των δραστηριοτήτων.

5. Εκμεταλλευτήκαμε την εύκολη και γρήγορη δυνατότητα οργάνωσης που προσφέρει το Trello ακόμα και για άτομα που δεν το έχουν χρησιμοποιήσει ξανά μία δεκάλεπτη παρακολούθηση στο Ιντερνέτ βίντεο επεξήγησης είναι αρκετή για να μάθει κάποιος τα βασικά εργαλεία του προγράμματος αυτού. Κατά τη χρήση του ήταν αρκετά βολικό στο να ανατρέχει κάποιος και να ενημερώνεται για την πρόοδο που έχει γίνει αλλά και για το ποια κομμάτια του έργου μένουν να υλοποιηθούν.

6. Δεν χρησιμοποιήσαμε κάποια σύνδεση με άλλες εφαρμογές.

# ΔΡΑΣΤΗΡΙΟΤΗΤΑ 2 - ΥΛΟΠΟΙΗΣΗ ΕΡΓΟΥ

## Ερώτημα Α– Διάγραμμα Κλάσεων και Υλοποίηση Κλάσεων σε Java

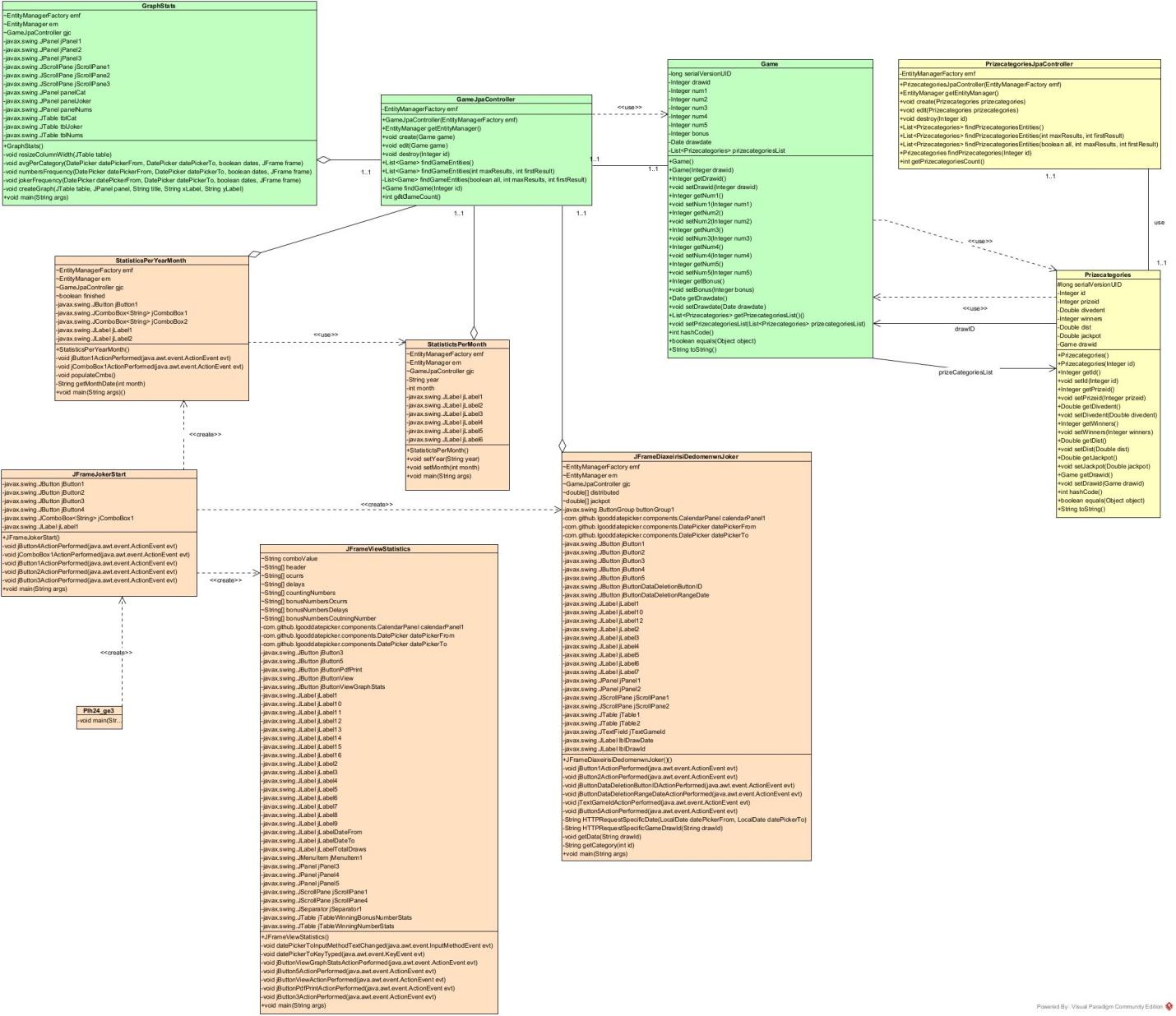
*[α) Εισάγεται το διάγραμμα κλάσεων,*

(β) το διάγραμμα Entity-Relationship (ER) του συστήματος,

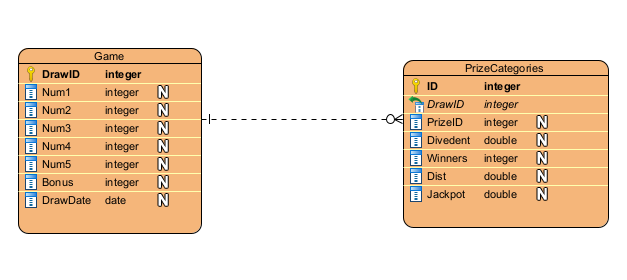
*(γ) Εισάγεται των κώδικα για τις κλάσεις του συστήματος και*

*(δ) Εισάγετε τον κώδικα java για τις pojoκλάσεις. Θα πρέπει να εισάγεται ΜΟΝΟ τον κώδικα που έχετε γράψει εσείς και όχι αυτόν που παράγεται αυτόματα προσδιορίζοντας ευκρινώς το όνομα της κλάσης και της μεθόδου****.*** *Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να υπάρχει τεκμηρίωση με μορφή σχολίων. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.]*

## α) Το διάγραμμα κλάσεων είναι το εξής:



## β) Το διάγραμμα Entity-Relationshipτου συστήματος είναι το εξής:

Στον πίνακα **Game** αποθηκεύονται οι κληρώσεις τζόκερ με κύριο κλειδί τον αριθμό της κλήρωσης (πεδίο **DrawID**) μαζί με τους 5 τυχερούς αριθμούς (πεδία **Num1**, **Num2**, **Num3**, **Num4**, **Num5**αντίστοιχα), τον αριθμό τζόκερ(πεδίο **Bonus**) καθώς και την ημερομηνία διεξαγωγής της κλήρωσης (πεδίο **DrawDate**).

Στον πίνακα **PrizeCategories** αποθηκεύονται τα στοιχεία που αφορούν τις κατηγορίες επιτυχιών κάθε κλήρωσης. Κύριο κλειδί του πίνακα είναι το πεδίο **ID** το οποίο ουσιαστικά είναι ένας αύξων αριθμός για κάθε εγγραφή και παράγεται αυτόματα. Το πεδίο **DrawID** είναι ξένο κλειδί από τον πίνακα **Game** για να ξέρουμε σε ποια κλήρωση αντιστοιχούν τα δεδομένα των κερδών. Τα υπόλοιπα πεδία:

* **PrizeID**: το idτης κάθε κατηγορίας κερδών (1=5+1, 2=5, 3=4+1 κ.ο.κ.)
* **Divedent**: Κέρδη ανά επιτυχία
* **Winners**: Νικητές στην κατηγορία
* **Dist**: Απότοdistributed, ταδιανεμόμενα κέρδη κάθε κατηγορίας
* **Jackpot**: Το ποσό χρημάτων που προστίθεται στα διανεμόμενα κέρδη της 1ης κατηγορία επιτυχιών σε περίπτωση που δεν υπάρχει νικητής.

Ο πίνακας **Game** συνδέεται με τον πίνακα **PrizeCategories** μέσω του πεδίου **DrawID** με σχέση 1 προς πολλά.

Το όνομα της βάσης δεδομένων είναι **JOKER** και βρίσκεται αποθηκευμένη στον φάκελο του projectστον υποφάκελο **\database**.Username:**JOKER**Password: **JOKER**

**SQL Κώδικας για την δημιουργία της βάσης δεδομένων**

**CREATETABLE** Game **(**DrawID **integerNOTNULL,** Num1 **integer,** Num2 **integer,** Num3 **integer,** Num4 **integer,** Num5 **integer,** Bonus **integer,** DrawDate **date,PRIMARYKEY(**DrawID**));**

**CREATETABLE** PrizeCategories **(**ID **integer** GENERATED **BYDEFAULTASIDENTITY,** DrawID **integerNOTNULL,** PrizeID **integer,** Divedent **double,** Winners **integer,** Dist **double,** Jackpot **double,PRIMARYKEY(**ID**));**

**ALTERTABLE** PrizeCategories **ADDCONSTRAINT** FKPrizeCateg466330 **FOREIGNKEY(**DrawID**)REFERENCES** Game **(**DrawID**);**

*Μαζί παραδίδεται και το αρχείο* ***db\_creation\_script.sql*** *που περιέχει τον παραπάνω κώδικα.*

## γ) Κώδικας με τις κλάσεις του συστήματος

Η κλάση main της εφαρμογής:

**Plh24\_ge3.java**package plh24\_ge3;  
  
/\*\*  
 \*  
 \* @author Manolis  
 \*/  
publicclass Plh24\_ge3 {  
  
/\*\*  
 \* @param args the command line arguments  
 \*/  
publicstaticvoid main(String[] args) {  
//Άνοιγματουκυρίωςμενούτηςεφαρμογής  
 JFrameJokerStart jokerStart = new JFrameJokerStart();  
 jokerStart.pack();  
 jokerStart.setLocationRelativeTo(null);  
 jokerStart.setVisible(true);  
 }  
}

## δ) Κώδικας από τις pojo κλάσεις:

**Game.java**

//NamedQuery για την αναζήτηση κληρώσεων σε εύρος ημερομηνιών  
 , @NamedQuery(name = "Game.findBetweenDrawdate", query = "SELECT g FROM Game g WHERE g.drawdate BETWEEN :drawdateFrom AND :drawdateTo")

**Prizecategories.java**  
//NamedQuery για αναζήτηση κατηγοριών κερδών με βάση το id της κλήρωσης  
 , @NamedQuery(name = "Prizecategories.findByDrawId", query = "SELECT p FROM Prizecategories p WHERE p.drawid = :drawid")

## Ερώτημα Β – Δημιουργία GUI Εφαρμογής

*[Εισάγετε τον κώδικα java για τις σχετικές κλάσεις. Θα πρέπει να εισάγεται ΜΟΝΟ τον κώδικα που έχετε γράψει εσείς και όχι αυτόν που παράγεται αυτόματα προσδιορίζοντας ευκρινώς το όνομα της κλάσης και της μεθόδου Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να υπάρχει τεκμηρίωση με μορφή σχολίων. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.]*

*Όλος ο κώδικας αναπτύχθηκε μέσα στις GUI κλάσεις, επομένως στο επόμενο ερώτημα (3.3) θα παρατεθεί ο κώδικας των κλάσεων για τις απαιτήσεις R4 και R5 καθώς αυτές θεωρούμε ότι παρουσιάζουν «στατιστικά στοιχεία», παρόλο που οι κλάσεις είναι GUI.*

**JFrameJokerStart.java**

Η κλάση που περιέχει το αρχικό μενού της εφαρμογής (απαίτηση R1).

/\*\*  
 \* Creates new form JFrameJokerStart  
 \*/  
**public JFrameJokerStart()**{  
this.setTitle("ΠαιχνίδιαΟΠΑΠ");  
 initComponents();  
 }

**privatevoid jButton4ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
if (JOptionPane.showConfirmDialog(null, "Έξοδοςαπότηνεφαρμογή;", "Έξοδος",  
 JOptionPane.YES\_NO\_OPTION) == JOptionPane.YES\_OPTION) {  
//Κατέστρεψετοπαράθυρο  
 dispose();  
//Τερμάτισεμεκωδικόεξόδου 0   
 System.exit(0);  
 }  
 }

**privatevoid jComboBox1ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
 jButton1.setText("1. Διαχείρισηδεδομένων " + jComboBox1.getSelectedItem());  
 jButton2.setText("2. Προβολήδεδομένων " + jComboBox1.getSelectedItem() + " ανάμήναγιασυγκεκριμένοέτος");  
 jButton3.setText("3. Προβολήστατιστικώνδεδομένων " + jComboBox1.getSelectedItem() + " καιεκτύπωσησεαρχείοpdf");  
 }

**privatevoidjButton1ActionPerformed**(java.awt.event.ActionEventevt) {   
// TODOaddyourhandlingcodehere:  
if (jComboBox1.getSelectedItem() == "ΤΖΟΚΕΡ") {  
JFrameDiaxeirisiDedomenwnJokerJFDedomenwnJoker = newJFrameDiaxeirisiDedomenwnJoker();  
JFDedomenwnJoker.setTitle("ΔιαχείρισηδεδομένωνΤΖΟΚΕΡ");  
JFDedomenwnJoker.pack();  
JFDedomenwnJoker.setLocationRelativeTo(null);  
JFDedomenwnJoker.setVisible(true);  
//JFDedomenwnJoker.setAlwaysOnTop(true);  
 } else {  
JOptionPane.showMessageDialog(null, "δενέχειυλοποιηθείακόμα...θαυλοποιηθείσύντομα");  
 }  
 }

**privatevoidjButton2ActionPerformed**(java.awt.event.ActionEventevt) {   
// TODOaddyourhandlingcodehere:  
if (jComboBox1.getSelectedItem() == "ΤΖΟΚΕΡ") {  
//Άνοιγμαφόρμαςκαιεμφάνισηπαραθύρουστοκέντροτηςοθόνης  
StatisticsPerYearMonthspm = newStatisticsPerYearMonth();  
spm.pack();  
spm.setLocationRelativeTo(null);  
spm.setVisible(true);  
 } else {  
JOptionPane.showMessageDialog(null, "δενέχειυλοποιηθείακόμα...θαυλοποιηθείσύντομα");  
 }  
 }

**privatevoidjButton3ActionPerformed**(java.awt.event.ActionEventevt) {   
// TODOaddyourhandlingcodehere:  
if (jComboBox1.getSelectedItem() == "ΤΖΟΚΕΡ") {  
JFrameViewStatisticsvs = newJFrameViewStatistics();  
vs.pack();  
vs.setLocationRelativeTo(null);  
vs.setVisible(true);  
 } else {  
JOptionPane.showMessageDialog(null, "δενέχειυλοποιηθείακόμα...θαυλοποιηθείσύντομα");  
 }  
 }

**JFrameDiaxeirisiDedomenwnJoker.java**

Κλάση από στην οποία ο χρήστης μπορεί να αναζητήσει κληρώσεις, να τις αποθηκεύσει στην βάση δεδομένων καθώς και να τις διαγράψει (απαίτηση R2).

/\*\*  
 \* Creates new form NewJFrame  
 \*/  
**public JFrameDiaxeirisiDedomenwnJoker()**{  
 initComponents();  
 DatePickerSettings dateSettings = new DatePickerSettings();  
 dateSettings.setFormatForDatesCommonEra("dd/MM/yyyy");  
 datePickerFrom.setDateToToday();  
 datePickerFrom.setSettings(dateSettings);  
  
 DatePickerSettings dateSettings2 = new DatePickerSettings();  
 dateSettings2.setFormatForDatesCommonEra("dd/MM/yyyy");  
 datePickerTo.setDateToToday();  
 datePickerTo.setSettings(dateSettings2);  
  
 lblDrawDate.setText("");  
 lblDrawId.setText("");  
 }

**privatevoid jButton1ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
if (!jTextGameId.getText().isEmpty()) {  
 getData(jTextGameId.getText());  
 } else {  
 JOptionPane.showMessageDialog(null, "Εισάγετεαριθμόκλήρωσηςστοπεδίο", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
 }  
 }   
  
**privatevoid jButton2ActionPerformed**(java.awt.event.ActionEvent evt) {   
  
if (lblDrawId.getText().isEmpty() || lblDrawDate.getText().isEmpty()) {  
 JOptionPane.showMessageDialog(null, "Δενυπάρχουνδεδομέναγιααποθήκευση!", this.getTitle(), JOptionPane.ERROR\_MESSAGE);  
return;  
 }  
  
 SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");  
 Date parsed = null;  
try {  
 parsed = format.parse(lblDrawDate.getText());  
 } catch (ParseException ex) {  
 Logger.getLogger(JFrameDiaxeirisiDedomenwnJoker.class.getName()).log(Level.SEVERE, null, ex);  
 }  
 java.sql.Date sqlDrawDate = new java.sql.Date(parsed.getTime());  
  
final String connString = "jdbc:derby:";  
final String dbPath = "//localhost:1527/JOKER";  
final String user = "JOKER";  
final String pass = "JOKER";  
  
 DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();  
int nCol = dtm.getColumnCount();  
int winningNumbers[] = newint[nCol];  
for (int i = 0; i < nCol; i++) {  
 winningNumbers[i] = Integer.parseInt(dtm.getValueAt(0, i).toString());  
 }  
  
 DefaultTableModel dtm2 = (DefaultTableModel) jTable2.getModel();  
int nRow = dtm2.getRowCount();  
 nCol = dtm2.getColumnCount();  
 String[][] prizeCategories = new String[nRow][nCol - 1]; //-1 στιςστήλεςγιατίδενθέλουμετην 1ηστήληπουπεριέχειταονόματατωνκατηγοριών  
for (int i = 0; i < nRow; i++) {  
for (int j = 0; j < nCol - 1; j++) {  
 prizeCategories[i][j] = dtm2.getValueAt(i, j + 1).toString();//+1 γιατονίδιολόγομεπαραπάνω  
 }  
 }  
  
int drawId = Integer.parseInt(lblDrawId.getText());  
 String SELECT\_QUERY = "SELECT DRAWID FROM JOKER.GAME WHERE DRAWID = " + drawId;  
  
boolean success = false;  
try {  
 Connection con = DriverManager.getConnection(connString + dbPath, user, pass); // ΣυνδεσημετηΒάσηδεδομένων  
 Statement stmt = con.createStatement();  
  
 ResultSet result = stmt.executeQuery(SELECT\_QUERY);  
  
if (result.next()) {  
 success = true;  
 JOptionPane.showMessageDialog(null, "Ηκλήρωση " + drawId + " είναιήδηαποθηκευμένηστηνβάσηδεδομένων", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
 } else {  
  
 PreparedStatement psG = con.prepareStatement("INSERT INTO JOKER.GAME VALUES(?,?,?,?,?,?,?,?)");  
 psG.setInt(1, drawId);  
 psG.setInt(2, winningNumbers[0]);  
 psG.setInt(3, winningNumbers[1]);  
 psG.setInt(4, winningNumbers[2]);  
 psG.setInt(5, winningNumbers[3]);  
 psG.setInt(6, winningNumbers[4]);  
 psG.setInt(7, winningNumbers[5]);  
 psG.setDate(8, sqlDrawDate);  
 psG.executeUpdate();  
  
for (int i = 0; i < 8; i++) {  
 PreparedStatement psP = con.prepareStatement("INSERT INTO JOKER.PRIZECATEGORIES (DRAWID,PRIZEID,DIVEDENT,WINNERS,DIST,JACKPOT) VALUES (?,?,?,?,?,?)");  
  
 psP.setInt(1, drawId);  
 psP.setInt(2, i + 1);  
  
 NumberFormat nFormat = NumberFormat.getInstance(Locale.getDefault());  
 Number number = nFormat.parse(prizeCategories[i][1]);  
double d = number.doubleValue();  
 psP.setDouble(3, d);  
  
 number = nFormat.parse(prizeCategories[i][0]);  
int integer = number.intValue();  
 psP.setInt(4, integer);  
  
 psP.setDouble(5, distributed[i]);  
 psP.setDouble(6, jackpot[i]);  
 psP.executeUpdate();  
 }  
  
 }  
 } catch (SQLException e) {  
 success = true;  
 System.out.println(e);  
 JOptionPane.showMessageDialog(null, e + "\nΤαδεδομέναδεναποθηκέυτηκαν", this.getTitle(), JOptionPane.ERROR\_MESSAGE);  
 } catch (ParseException ex) {  
 success = true;  
 System.out.println(ex);  
 JOptionPane.showMessageDialog(null, ex + "\nΤαδεδομέναδεναποθηκέυτηκαν", this.getTitle(), JOptionPane.ERROR\_MESSAGE);  
 }  
  
if (!success) {  
 JOptionPane.showMessageDialog(null, "Ταδεδομένααποθηκεύτηκανμεεπιτυχία", this.getTitle(), JOptionPane.INFORMATION\_MESSAGE);  
 }  
  
 }   
  
**privatevoid jButtonDataDeletionButtonIDActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
 JFrame frame = new JFrame("ΔιαγραφήΔεδομένωνΠαιχνιδιού");  
 frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 JPanel panel = new JPanel();  
 panel.setLayout(null);  
  
// Δημιουργία Combobox  
 JComboBox c1 = new JComboBox();  
 Dimension size1 = c1.getPreferredSize();  
  
 c1.setBounds(170, 20, 60, size1.height);  
//Προσθήκηστο panel του frame  
 panel.add(c1);  
  
//Επιλογήτων drawId's πουυπάρχουνστηνβάσηδεδομένων  
//ωστεναεπιλέξειοχρήστηςποίακλήρωσηθέλειναδιαγράψει  
 List<Integer> drawIds = new ArrayList<>();  
try {  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findAll");  
  
 List<Game> games = new ArrayList<>();  
 games = query.getResultList();  
 em.getTransaction().commit();  
  
for (Game game : games) {  
int drawId = game.getDrawid();  
 drawIds.add(drawId);  
 }  
  
 drawIds.forEach((drawId) -> {  
 c1.addItem(String.valueOf(drawId));  
 });  
 } catch (Exception ex) {  
 JOptionPane.showMessageDialog(this, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 }  
  
// Δημιουργία Button  
 JButton button = new JButton("Διαγραφήδεδομένωνπαιχνιδιού");  
 Dimension size2 = button.getPreferredSize();  
 button.setBounds(160, 80, size2.width, size2.height);  
//ActionListener γιατο Button  
 button.addActionListener((ActionEvent e) -> {  
int drawId = Integer.parseInt(c1.getSelectedItem().toString());  
//Ερώτησηστονχρήστηπριντηνδιαγραφή  
if (JOptionPane.showConfirmDialog(null, "Θέλετεναδιαγράψετετηνκλήρωση " + drawId + ";", "ΔιαγραφήΔεδομένων", JOptionPane.YES\_NO\_OPTION) == JOptionPane.NO\_OPTION) {  
return;  
 }  
try {  
 em.getTransaction().begin();  
//Πρώταδιαγραφήτωνδεδομένωντηςκλήρωσηςαπότονπίνακα PRIZECATEGORIES  
 Game gameDrawId = gjc.findGame(drawId);  
 Query query2 = em.createQuery("DELETE FROM Prizecategories WHERE drawid = :drawid");  
 query2.setParameter("drawid", gameDrawId).executeUpdate();  
  
//Καιμετάδιαγραφήτηςκλήρωσηςαπότονπίνακα GAME   
 Query query3 = em.createQuery("DELETE FROM Game WHERE gameid = :drawid");  
 query3.setParameter("drawid", drawId).executeUpdate();  
 em.getTransaction().commit();  
 JOptionPane.showMessageDialog(this, "Διαγράφηκεηκλήρωσημεαριθμο: " + drawId);  
 c1.removeItem(String.valueOf(drawId));  
 } catch (Exception ex) {  
 JOptionPane.showMessageDialog(this, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 System.out.println(ex);  
 }  
 });  
//Προσθήκηστο panel του frame  
 panel.add(button);  
  
 frame.getContentPane().add(panel, BorderLayout.CENTER);  
 frame.setSize(400, 200);  
 frame.setResizable(false);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }   
  
**privatevoid jButtonDataDeletionRangeDateActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
 JFrame frame = new JFrame("Διαγραφήδεδομένωνπαιχνιδιώνεντόςεύρουςημερομηνιών");  
 frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 JPanel panel = new JPanel();  
 panel.setLayout(null);  
  
//Δημιουργία Label  
 JLabel lblFrom = new JLabel();  
 lblFrom.setText("Από:");  
 Dimension size = datePickerFrom.getPreferredSize();  
 lblFrom.setBounds(1, 10, size.width, size.height);  
 panel.add(lblFrom);  
  
// Δημιουργία DatePicker from  
 DatePicker datePickerFrom = new DatePicker();  
 DatePickerSettings dateSettings = new DatePickerSettings();  
 Dimension size1 = datePickerFrom.getPreferredSize();  
 datePickerFrom.setBounds(30, 10, size1.width, size1.height);  
 dateSettings.setFormatForDatesCommonEra("dd/MM/yyyy");  
 datePickerFrom.setDateToToday();  
 datePickerFrom.setSettings(dateSettings);  
//Προσθήκηστο panel του frame  
 panel.add(datePickerFrom);  
  
//Δημιουργία Label  
 JLabel lblTo = new JLabel();  
 lblTo.setText("Έως:");  
 Dimension size0 = datePickerFrom.getPreferredSize();  
 lblTo.setBounds(200, 10, size0.width, size0.height);  
 panel.add(lblTo);  
  
// Δημιουργία DatePicker to  
 DatePicker datePickerTo = new DatePicker();  
 DatePickerSettings dateSettings2 = new DatePickerSettings();  
 Dimension size2 = datePickerTo.getPreferredSize();  
 datePickerTo.setBounds(232, 10, size2.width, size2.height);  
 dateSettings2.setFormatForDatesCommonEra("dd/MM/yyyy");  
 datePickerTo.setDateToToday();  
 datePickerTo.setSettings(dateSettings2);  
//Προσθήκηστο panel του frame  
 panel.add(datePickerTo);  
  
// Δημιουργία Button  
 JButton button = new JButton("Διαγραφήδεδομένωνπαιχνιδιώνεντόςεύρουςημερομηνιών");  
 Dimension size3 = button.getPreferredSize();  
 button.setBounds(15, 80, size3.width, size3.height);  
//ActionListener γιατο Button  
 button.addActionListener((ActionEvent e) -> {  
if (JOptionPane.showConfirmDialog(null, "Θέλετεναδιαγράψετετιςκληρώσειςαπό " + datePickerFrom.getDateStringOrEmptyString()  
 + " μέχρι " + datePickerTo.getDateStringOrEmptyString() + ";", "ΔιαγραφήΔεδομένων", JOptionPane.YES\_NO\_OPTION) == JOptionPane.NO\_OPTION) {  
return;  
 }  
  
try {  
//Παίρνωτοεύροςτωνημερομηνιώναπότα controls datePicker  
 DateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
 java.util.Date dateFrom = format.parse(datePickerFrom.getDateStringOrEmptyString());  
 java.util.Date dateTo = format.parse(datePickerTo.getDateStringOrEmptyString());  
  
//Βρίσκωτα drawIds τωνκληρώσεων, μεταξύτωνημερομηνιώνπουέδωσεοχρήστης,  
//καιτααποθηκεύωστηνλίστααντικειμένων games  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findBetweenDrawdate");  
 query.setParameter("drawdateFrom", dateFrom);  
 query.setParameter("drawdateTo", dateTo);  
//λίστααντικειμένων games  
 List<Game> games = new ArrayList<>();  
 games = query.getResultList();  
 em.getTransaction().commit();  
  
//Γιακάθε drawId πουβρέθηκεδιάγραφωτηναντίστοιχααποτελέσματααπότονπίνακα PRIZECATEGORIES  
//καιμετατηνκλήρωσηαπότονπίνακα GAME  
for (Game game : games) {  
 em.getTransaction().begin();  
 Game gameDrawId = gjc.findGame(game.getDrawid());  
  
//Διαγραφήαπότονπίνακα PRIZECATEGORIES πρώτα  
 Query query1 = em.createQuery("DELETE FROM Prizecategories WHERE drawid = :drawid");  
 query1.setParameter("drawid", gameDrawId).executeUpdate();  
  
//Καιμετάδιαγραφήαπότονπίνακα GAME  
 Query query2 = em.createQuery("DELETE FROM Game WHERE drawid = :drawid");  
 query2.setParameter("drawid", game.getDrawid()).executeUpdate();  
 em.getTransaction().commit();  
 }  
  
//Εμφάνισηςμηνύματοςεπιτυχίαςδιαγραφήςστονχρήστη  
 JOptionPane.showMessageDialog(this, "Διαγράφηκανοικληρώσειςαπό " + datePickerFrom.getDateStringOrEmptyString() + " μέχρι " + datePickerTo.getDateStringOrEmptyString());  
 } catch (Exception ex) {  
 JOptionPane.showMessageDialog(this, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 System.out.println(ex);  
 }  
 });  
//Προσθήκηστο panel του frame  
 panel.add(button);  
  
 frame.getContentPane().add(panel, BorderLayout.CENTER);  
 frame.setSize(410, 200);  
 frame.setResizable(false);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }

**privatevoid jButton5ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
if (datePickerFrom.getText().equals("")) {  
 JOptionPane.showMessageDialog(null, "παρακαλώδηλώστεημερομηνίααπό...", this.getTitle(), JOptionPane.INFORMATION\_MESSAGE);  
 } elseif (datePickerTo.getText().equals("")) {  
 JOptionPane.showMessageDialog(null, "παρακαλώδηλώστεημερομηνίαεώς...", this.getTitle(), JOptionPane.INFORMATION\_MESSAGE);  
 } else {  
 LocalDate datePFrom = datePickerFrom.getDate();  
 LocalDate datePTo = datePickerTo.getDate();  
  
//εισαγωγήστοιχείουτου jcalendar γιασύγκρισηημερομηνιών  
long days = ChronoUnit.DAYS.between(datePFrom, datePTo);  
  
 String parseResponseString = HTTPRequestSpecificDate(datePFrom, datePTo);  
  
 GsonBuilder builder = new GsonBuilder();  
 builder.setPrettyPrinting();  
 Gson gson = builder.create();  
  
 gson.fromJson(parseResponseString, JsonArray.class);  
  
//έλεγχοςλίστας, εανείναι Null τότεδενυπάρχειπαιχνίδιγιατιςδιαθέσιμεςημερομηνίεςδιαφορετικάεκτελείταικανονικά  
if(days>=31){  
 JOptionPane.showMessageDialog(null, "Παρακαλώόχιπάνωαπό 31 ημέρες, μικρύνετετοεύροςημερομηνιών","ΠΡΟΣΟΧΗ!!",JOptionPane.INFORMATION\_MESSAGE);  
 }else{  
if (Objects.isNull(parseResponseString)) {  
 JOptionPane.showMessageDialog(null, "Δενυπάρχειδιαθέσιμοπαιχνίδι. Ελέγξτετιςημερομηνίες", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
 } else {  
//substring ώστενααφαιρέσωτιςαγκύλεςτηςπρώτηςκαιτελευαίαςαγκύληςτου String  
 parseResponseString = parseResponseString.substring(1, parseResponseString.length() - 1);  
  
//μετατροπήαπό string σε array  
 String[] elements = parseResponseString.split(",");  
  
//\*\*δημιουργίαπαραθύρου popup επιλογήςδιαθέσιμωνπαιχνιδιών\*\*//  
//δημιουργία panel και frame  
 JFrame frame = new JFrame("Επιλογήπαιχνιδιού");  
 JPanel panel = new JPanel();  
 panel.setLayout(new FlowLayout());  
//δημιουργία label, button και combobox  
 JLabel label = new JLabel("Παρακαλώεπιλέξτεπαιχνίτιαπότηλίστα: ");  
 JButton button = new JButton();  
 JComboBox combo = new JComboBox(elements);  
 button.setText("Επιλογή");  
//εισαγωγήστοιχείωνστο panel  
 panel.add(label);  
 panel.add(combo);  
 panel.add(button);  
//εισαγωγη panel στο frame καιρυθμίσειςτου frame  
 frame.add(panel);  
 frame.setSize(300, 300);  
 frame.setLocationRelativeTo(null);  
 frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 frame.setAlwaysOnTop(true); //always on top  
 frame.setVisible(true);  
  
//JButton action listener  
//απενεργοποιείτο popup καιστέλνειτηνεπιλογήτου combobox στο text game id  
 button.addActionListener(e -> {  
 getData(combo.getSelectedItem().toString());  
 frame.setAlwaysOnTop(false);  
 frame.setVisible(false);  
 });  
 }  
 }  
 }  
 }

//Δημιουργίαμεθόδου HTTP request συγκεκριμένηςημερομηνίας  
**private String HTTPRequestSpecificDate**(LocalDate datePickerFrom, LocalDate datePickerTo) {  
//Endeiktiko paradeigma EAP όπωςαναφέρεταιαπότηνδοκιμαστικήοδηγία.  
 String urlToCall = "https://api.opap.gr/draws/v3.0/5104/draw-date/" + datePickerFrom + "/" + datePickerTo + "/draw-id";  
 OkHttpClient client = new OkHttpClient();  
 Request request = new Request.Builder().url(urlToCall).build();  
  
try (Response response = client.newCall(request).execute()) {  
if (response.isSuccessful() && response.body() != null) {  
 String responseString = response.body().string();  
return responseString;  
  
 }  
 } catch (IOException e) {  
 System.out.println(e);  
returnnull;  
 }  
returnnull;  
 }  
  
//Δημιουργίαμεθόδου HTTP request συγκεκριμένουπαιχνιδιούβάση draw-id  
**private String HTTPRequestSpecificGameDrawId**(String drawId) {  
 String urlToCall = "https://api.opap.gr/draws/v3.0/5104/" + drawId;  
 OkHttpClient client = new OkHttpClient();  
 Request request = new Request.Builder().url(urlToCall).build();  
  
try (Response response = client.newCall(request).execute()) {  
if (response.isSuccessful() && response.body() != null) {  
 String responseString = response.body().string();  
return responseString;  
 }  
 } catch (IOException e) {  
 System.out.println(e);  
returnnull;  
 }  
returnnull;  
 }  
  
**privatevoid getData**(String drawId) {  
 String parseResponseString = HTTPRequestSpecificGameDrawId(drawId);  
//Έλεγχοςανυπάρχουνδεδομέναγιατοναριθμόκλήρωσηςπουέδωσεοχρήστης  
if (Objects.isNull(parseResponseString)) {  
 JOptionPane.showMessageDialog(null, "Δενυπάρχεικλήρωσημεαυτόντοναριθμό", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
return;  
 }  
  
//\*\*Εκκίνηση Gson Parsing\*\*//  
//http://tutorials.jenkov.com/java-json/gson.html  
//αρχικοίησηαντικειμένων Json  
 GsonBuilder builder = new GsonBuilder();  
 builder.setPrettyPrinting();  
 Gson gson = builder.create();  
  
//δημιουργίααντικειμένουμεχρήσητου HTTP request parseResponseString πουπήραμορφοποίησααπότημέθοδο HTTPRequestSpecificGameDrawId() χρησιμοποιώντας prety printing  
//βάσηοδηγιών GsonTester τουΕΑΠθαγίνειδημιουργίααντικειμένου JSOn ώστεναμπορέσωνακάνω list parsing/itteraing μέσωβιλβιοθηκώντηςμεθόδου  
 JsonObject jsonObj = gson.fromJson(parseResponseString, JsonObject.class);   
  
//Aπότοαντικείμενοτηςκλάσηςθαπάρωτοαντικείμενο object WinningNumbers object ώστεναχρησιμοποιήσωτηνλίστααυτού  
 JsonObject winningNumbersObject = jsonObj.getAsJsonObject("winningNumbers"); //winningNumber Object   
 JsonArray winningNumbersArray = winningNumbersObject.get("list").getAsJsonArray();//επιλογήλίστας winningNumberList from winningNumbersObject   
 JsonArray jokerNumberArray = winningNumbersObject.get("bonus").getAsJsonArray();//επιλογήλίστας winningNumberList from winningNumbersObject   
//Aπότοαντικείμενοτηςκλάσηςθαπάρωτοαντικείμενο array priZeCategories ώστεναχρησιμοποιήσωτηνλίστααυτού  
 JsonArray prizeCategoriesArray = jsonObj.get("prizeCategories").getAsJsonArray();   
  
//Μετατροπή Unix timestamp σε date  
long unixSeconds = (long) jsonObj.get("drawTime").getAsLong();  
 Date date = new java.util.Date(unixSeconds);  
 SimpleDateFormat sdf = new java.text.SimpleDateFormat("dd/MM/yyyy");  
 sdf.setTimeZone(java.util.TimeZone.getTimeZone("GMT+2"));  
 String drawDate = sdf.format(date);  
 lblDrawDate.setText(drawDate);   
  
// Δημιουργία Table Model πινάκων  
 DefaultTableModel dTable1 = (DefaultTableModel) jTable1.getModel();  
 DefaultTableModel dTable2 = (DefaultTableModel) jTable2.getModel();  
  
//Δημιουργίαλίσταςκαιταξινόμηση  
 Integer[] jTableListItems = new Integer[5];  
  
 dTable1.getDataVector().removeAllElements();//διαγραφήτωνδεδομένωντωνλιστώντουπίνακα jTable ώστεναμηναυξάνεταιαλλάναμηδενίζεισεκάθενέοπαιχνίδι  
 dTable1.fireTableDataChanged();//επιβεβαίωσηδιαγραφήςδεδομένων  
  
for (int i = 0; i < 5; i++) {  
 jTableListItems[i] = winningNumbersArray.get(i).getAsInt();  
 }  
 Arrays.sort(jTableListItems);  
  
//array copy  
 Integer[] jTableListItems2 = new Integer[6];  
for (int i = 0; i < jTableListItems.length; i++) {  
 jTableListItems2[i] = jTableListItems[i];  
 }  
 jTableListItems2[5] = jokerNumberArray.get(0).getAsInt();  
 dTable1.addRow(jTableListItems2);  
  
 String[] jTableListPrizeCategoryItems = new String[3];  
  
 dTable2.getDataVector().removeAllElements(); //διαγραφήτωνδεδομένωντωνλιστώντουπίνακα jTable ώστεναμηναυξάνεταιαλλάναμηδενίζεισεκάθενέοπαιχνίδι  
 dTable2.fireTableDataChanged();//επιβεβαίωσηδιαγραφήςδεδομένων  
  
  
int i = 0;  
for (JsonElement element : prizeCategoriesArray) {   
 jTableListPrizeCategoryItems[0] = getCategory(element.getAsJsonObject().get("id").getAsInt());  
 jTableListPrizeCategoryItems[1] = NumberFormat.getNumberInstance(Locale.getDefault()).format(element.getAsJsonObject().get("winners").getAsInt());  
 jTableListPrizeCategoryItems[2] = String.format(Locale.getDefault(), "%,.2f", element.getAsJsonObject().get("divident").getAsDouble());  
  
//Αποθήκευσηδιανεμόμενωνκερδώνκαιποσών jackpot σεπίνακες  
//ωστεναμπορούννααποθηκευτούνστηνβάση, σεπερίπτωσηπουτοεπιλέξειοχρήστης  
 distributed[i] = element.getAsJsonObject().get("distributed").getAsDouble();  
 jackpot[i] = element.getAsJsonObject().get("jackpot").getAsDouble();  
 i++;  
  
 dTable2.addRow(jTableListPrizeCategoryItems);  
 }  
 lblDrawId.setText(drawId);  
 }  
  
//Μέθοδοςγιαεπιστροφήτηςκατηγορίαςκερδών  
**private String getCategory**(int id) {  
switch (id) {  
case 1:  
return"5+1";  
case 2:  
return"5";  
case 3:  
return"4+1";  
case 4:  
return"4";  
case 5:  
return"3+1";  
case 6:  
return"3";  
case 7:  
return"2+1";  
case 8:  
return"1+1";  
default:  
return"";  
 }  
 }

**StatisticsPerYearMonth.java**

Ηκλάσηδίνειστονχρήστητηνδυνατότηταναεπιλέξειέναέτοςαπότιςκληρώσειςπουείναιαποθηκευμένεςστηβάσηδεδομένωνκαιβάσητουςέτουςπουθαεπιλέξειεμφανίζονταιοιμήνεςγιατουςοποίουςυπάρχουναποθηκευμένεςκληρώσειςστηνβάσηδεδομένων.Στηνσυνέχειααφούέχειεπιλέξειέτοςκαιμήναμπορείναδεισυγκεντρωτικάδεδομέναπουαφορούντιςκληρώσειςγιατονεπιλεγμένομήνατουςεπιλεγμένουέτους(απαίτησηR3).

EntityManagerFactory emf = Persistence.createEntityManagerFactory("plh24\_ge3PU");  
 EntityManager em = emf.createEntityManager();  
 GameJpaController gjc = new GameJpaController(emf);  
  
boolean finished = false; //μεταβλητήγιαναελέγξωανέχειτελειώσειηαρχικοποιήσητηςφόρμας  
  
/\*\*  
 \* Creates new form StatisticsPerMonth  
 \*/  
**public StatisticsPerYearMonth()**{  
  
//Listener ώστε να ξεκινήσω την διαδικασία όταν το παράθυρο θα έχει ανοίξει  
this.addWindowListener(new WindowAdapter() {  
// Invoked when a window has been opened.  
 @Override  
publicvoid windowOpened(WindowEvent e) {  
 populateCmbs();  
 finished = true;  
 }  
 });  
  
this.setTitle("Προβολή δεδομένων ΤΖΟΚΕΡ ανά μήνα για συγκεκριμένο έτος");//Τίτλος της φόρμας  
 initComponents();  
 }

**privatevoid jButton1ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:   
 StatistictsPerMonth spm = new StatistictsPerMonth();  
 spm.setYear(jComboBox1.getSelectedItem().toString()); //Περνάω το έτος που επέλεξε ο χρήστης στην επόμενη οθόνη  
  
//Περνάω τον αριθμό του μήνα που επέλεξε ο χρήστης στην επόμενη οθόνη  
 Date date = null;  
try {  
 date = new SimpleDateFormat("MMMM", Locale.getDefault()).parse(jComboBox2.getSelectedItem().toString());  
 } catch (ParseException ex) {  
 Logger.getLogger(StatisticsPerYearMonth.class.getName()).log(Level.SEVERE, null, ex);  
 }  
 Calendar cal = Calendar.getInstance();  
 cal.setTime(date);  
  
 spm.setMonth(cal.get(Calendar.MONTH) + 1);  
 spm.pack();  
 spm.setLocationRelativeTo(null);  
 spm.setVisible(true);  
 }   
  
**privatevoid jComboBox1ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
if (!finished) {  
return;  
 }  
 populateCmbs();  
 }   
  
//Γέμισμα των comboboxes  
**privatevoid populateCmbs()**{  
try {  
//Επιλέγω όλες τις εγγραφές από τον πίνακα GAME  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findAll");  
  
//λίστα αντικειμένων games  
 List<Game> games = new ArrayList<>();  
 games = query.getResultList();  
 em.getTransaction().commit();  
  
 games.sort(Comparator.comparing(Game::getDrawdate));//Ταξινομώ την λίστα με τα αποτελέσματα από το query με βάση τις ημερομηνίες  
  
//Λίστα που θα αποθηκευθούν οι χρονιές για τις οποίες έχουμε αποθηκευμένα δεδομένα  
//Χρησιμοποιώ LinkedHashSet καθώς δεν επιτρέπει τα διπλότυπα  
 LinkedHashSet<Integer> years = new LinkedHashSet<>();  
for (Game game : games) {  
//Αποθηκεύω στην LinkedHashSet μόνο τις χρονιές  
 Date date = game.getDrawdate();  
 Calendar cal = Calendar.getInstance(TimeZone.getDefault());  
 cal.setTime(date);  
 years.add(cal.get(Calendar.YEAR));  
 }  
  
//Προσθήκη των ετών στο comboBox πριν αρχικοποιηθεί η φόρμα ΜΟΝΟ  
if (!finished) {  
 jComboBox1.removeAllItems();  
 years.forEach((year) -> {  
 jComboBox1.addItem(String.valueOf(year));  
 });  
 }  
  
//Προσθήκη μηνών στο 2ο Combobox αφού έχει επιλεχθεί η χρονία στο 1ο combobox  
 LinkedHashSet<Integer> months = new LinkedHashSet<>();  
for (Game game : games) {  
//Αποθηκεύω στην LinkedHashSet μόνο τις χρονιές  
 Date date = game.getDrawdate();  
 Calendar cal = Calendar.getInstance(TimeZone.getDefault());  
 cal.setTime(date);  
if (cal.get(Calendar.YEAR) == Integer.parseInt(jComboBox1.getSelectedItem().toString())) {  
 months.add(cal.get(Calendar.MONTH));  
 }  
 }  
 jComboBox2.removeAllItems();  
 months.forEach((month) -> {  
 jComboBox2.addItem(getMonthDate(month));  
 });  
 } catch (NumberFormatException ex) {  
 JOptionPane.showMessageDialog(null, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 }  
 }  
  
**private String getMonthDate**(int month) {  
switch (month) {  
case 0:  
return"Ιανουάριος";  
case 1:  
return"Φεβρουάριος";  
case 2:  
return"Μάρτιος";  
case 3:  
return"Απρίλιος";  
case 4:  
return"Μάιος";  
case 5:  
return"Ιούνιος";  
case 6:  
return"Ιούλιος";  
case 7:  
return"Αύγουστος";  
case 8:  
return"Σεπτέμβριος";  
case 9:  
return"Οκτώβριος";  
case 10:  
return"Νοέμβριος";  
case 11:  
return"Δεκέμβριος";  
default:  
return"";  
 }  
 }

**StatistictsPerMonth.java**

Η κλάση εμφανίζει στον χρήστη συγκεντρωτικά, για τον μήνα του έτους που επέλεξε τα εξής:

Πόσα παιχνίδια έγιναν

Πόσα χρήματα διανεμήθηκαν

Πόσα ΤΖΑΚ-ΠΟΤ έγιναν

(απαίτηση R3).

EntityManagerFactory emf = Persistence.createEntityManagerFactory("plh24\_ge3PU");  
 EntityManager em = emf.createEntityManager();  
 GameJpaController gjc = new GameJpaController(emf);  
private String year; //Το έτος που έχει επιλεχθεί στο προηγούμενο παράθυρο  
privateint month;//Ο μήνας που έχει επιλεχθεί στο προηγούμενο παράθυρο  
  
/\*\*  
 \* Creates new form StatistictsPerMonth  
 \*/  
**public StatistictsPerMonth()**{  
this.setTitle("Προβολή δεδομένων ανά μήνα");//Τίτλος της φόρμας  
 initComponents();  
  
//Listener ώστε να ξεκινήσω την διαδικασία όταν το παράθυρο θα έχει ανοίξει  
this.addWindowListener(new WindowAdapter() {  
// Invoked when a window has been opened.  
 @Override  
publicvoid windowOpened(WindowEvent e) {  
try {  
//Δημιουργώ την ημερομηνία από την οποία θα αρχίσει η ανάζητηση του sql query  
 SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
 Date parsed = format.parse(year + "-" + month + "-" + "01");  
 java.sql.Date dateFrom = new java.sql.Date(parsed.getTime());  
  
//Δημιουργώ την ημερομηνία στην οποία θα τελειώνει η ανάζητηση του sql query  
 parsed = format.parse(year + "-" + month + "-" + "31");  
 java.sql.Date dateTo = new java.sql.Date(parsed.getTime());  
  
//Επιλέγω όλες τις εγγραφές από τον πίνακα GAME που ανήκουν χρονικά στο έτος και στον μήνα που έχει επιλέξει ο χρήστης  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findBetweenDrawdate");  
 query.setParameter("drawdateFrom", dateFrom);  
 query.setParameter("drawdateTo", dateTo);  
  
//λίστα αντικειμένων games του έτους και του μήνα που ζητήθηκε από τον χρήστη  
 List<Game> games = new ArrayList<>();  
 games = query.getResultList();  
 em.getTransaction().commit();  
  
 jLabel4.setText(String.valueOf(games.size()));  
int jackpots = 0;  
double dist = 0;  
for (Game game : games) {  
 Game gameDrawId = gjc.findGame(game.getDrawid());  
 em.getTransaction().begin();  
 Query query2 = em.createNamedQuery("Prizecategories.findByDrawId");  
 query2.setParameter("drawid", gameDrawId);  
  
//-----------------------λίστα αντικειμένων games του έτους και του μήνα που ζητήθηκε από τον χρήστη  
 List<Prizecategories> prizeCategories = new ArrayList<>();  
 prizeCategories = query2.getResultList();  
 em.getTransaction().commit();  
  
for (Prizecategories prizeCategorie : prizeCategories) {  
if (prizeCategorie.getPrizeid() == 1 && prizeCategorie.getWinners() == 0) {  
 jackpots++;  
 }  
  
if (prizeCategorie.getPrizeid() == 1 && prizeCategorie.getWinners() > 0) {  
 dist += prizeCategorie.getDist() + prizeCategorie.getJackpot();  
 }   
  
if (prizeCategorie.getPrizeid() == 2 && prizeCategorie.getWinners() > 0) {  
 dist += prizeCategorie.getDist();  
 }  
  
if (prizeCategorie.getPrizeid() != 1 && prizeCategorie.getPrizeid() != 2) {  
 dist += prizeCategorie.getDist();  
 }   
 }  
 }  
  
 jLabel5.setText(String.format(String.format(Locale.getDefault(), "%,.2f",dist))+" €");  
 jLabel6.setText(String.valueOf(jackpots));  
 } catch (Exception ex) {  
 System.out.println(ex);  
 }  
 }  
 });  
  
 }  
  
//Setter για το έτος, ο οποίος καλείται στην προηγούμενη οθόνη  
**publicvoid setYear**(String year) {  
this.year = year;  
 }  
  
//Setter για τον μήνα, ο οποίος καλείται στην προηγούμενη οθόνη  
**publicvoid setMonth**(int month) {  
this.month = month;  
 }

## Ερώτημα Γ – Παρουσίαση Στατιστικών Στοιχείων και Γραφημάτων

*[Εισάγετε τον κώδικα javaπου αναφέρεται στην παρουσίαση των στατιστικών στοιχείων. Θα πρέπει να προσδιορίζεται ευκρινώς το όνομα της κλάσης και της μεθόδου. Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να υπάρχει τεκμηρίωση με μορφή σχολίων. Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.]*

**JFrameViewStatistics.java**

Η κλάση εμφανίζει στατιστικά των αριθμών που έχουν κληρωθεί με κλήση στο api<https://api.opap.gr/games/v1.0/5104/statistics>. Η απαίτηση για τα στατιστικά αριθμών εντός εύρους ημερομηνιών έγινε χρήση του api[https://api.opap.gr/draws/v3.0/{gameId}/draw-id/{fromDrawId}/{toDrawId}](https://api.opap.gr/draws/v3.0/%7bgameId%7d/draw-id/%7bfromDrawId%7d/%7btoDrawId%7d)χωρίς να εμφανίζονται όμως τον πλήθος των κληρώσεων που δεν εμφανίστικε κάποιος αριθμός (Delay). Επίσης δίνεται στον χρήστη η δυνατότητα να αποθηκεύσει σε αρχείο pdfτα αποτελέσματα (απαίτηση R4).

***Το αρχείο αποθηκεύεται στον φάκελο του project.***

//δημιουργία global μεταβλητών  
 String comboValue;  
  
// δημιουργίαπινάκωναποθήκευσηςδεδομένων  
 String[] header = new String[3];  
  
 String[] ocurrs = new String[50];  
 String[] delays = new String[50];  
 String[] countingNumbers = new String[50];  
  
 String[] bonusNumbersOcurrs = new String[50];  
 String[] bonusNumbersDelays = new String[50];  
 String[] bonusNumbersCoutningNumber = new String[50];  
  
/\*\*  
 \* Creates new form NewJFrame  
 \*/  
**public JFrameViewStatistics()**{

this.setTitle("Προβολή στατιστικών δεδομένων ΤΖΟΚΕΡ και εκτύπωση σε αρχείο pdf");   
initComponents();  
 }

**privatevoid jButtonViewGraphStatsActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
 GraphStats gs = new GraphStats();  
 gs.pack();  
 gs.setLocationRelativeTo(null);  
 gs.setVisible(true);  
 }   
  
**privatevoid jButton5ActionPerformed**(java.awt.event.ActionEvent evt) {   
// TODO add your handling code here:  
//υλοποίησηεπιλογήςπαιχνιδιώνκαιέξοδοςστατιστικώνβάσηεπιλο΄γηςημερομηνιών  
  
int SizeOfArrayWinningNumbers = 0; //αρχικοποίησημεταβλητήςγιαχρήσειςεκτός if..else ώστεναγίνειέλεγχοςκενήςλίστας  
  
if (datePickerFrom.getText().equals("")) {  
 JOptionPane.showMessageDialog(null, "παρακαλώδηλώστεημερομηνίααπό...", "ΠΡΟΣΟΧΗ!!", JOptionPane.INFORMATION\_MESSAGE);  
 } elseif (datePickerTo.getText().equals("")) {  
 JOptionPane.showMessageDialog(null, "παρακαλώδηλώστεημερομηνίαεώς...", "ΠΡΟΣΟΧΗ!!", JOptionPane.INFORMATION\_MESSAGE);  
 } else {  
  
 LocalDate datePFrom = datePickerFrom.getDate();  
 LocalDate datePTo = datePickerTo.getDate();  
  
//προτροπήμεμήνυμαστονχρήστηναμηνβάλειπάνωαπόέναόριοημερομηνιώνδιότισκάειτο API  
long days = ChronoUnit.DAYS.between(datePFrom, datePTo);  
if (days >= 25) {  
 JOptionPane.showMessageDialog(null, "Παρακαλώόχιπάνωαπό 25 ημέρες", "ΠΡΟΣΟΧΗ!!", JOptionPane.INFORMATION\_MESSAGE);  
return;  
 }  
  
//Endeiktiko paradeigma EAP όπωςαναφέρεταιαπότηνδοκιμαστικήοδηγία.  
 String urlToCall = "https://api.opap.gr/draws/v3.0/5104/draw-date/" + datePickerFrom + "/" + datePickerTo;  
 OkHttpClient client = new OkHttpClient();  
 Request request = new Request.Builder().url(urlToCall).build();  
 String parseResponseString = null;  
try (Response response = client.newCall(request).execute()) {  
if (response.isSuccessful() && response.body() != null) {  
 parseResponseString = response.body().string();  
 }  
 } catch (IOException e) {  
 System.out.println(e);  
 }  
 GsonBuilder builder = new GsonBuilder();  
 builder.setPrettyPrinting();  
 Gson gson = builder.create();  
  
 JsonObject jsonObj = gson.fromJson(parseResponseString, JsonObject.class);  
  
 jTableWinningNumberStats.setVisible(true);  
 jTableWinningBonusNumberStats.setVisible(true);  
this.jButtonPdfPrint.setEnabled(true);  
  
 JsonPrimitive prim = (JsonPrimitive) jsonObj.get("totalElements");//πήρατο totalElements  
  
 SizeOfArrayWinningNumbers = prim.getAsInt();  
int SizeOfArrayBonusNumbers = prim.getAsInt();  
  
//το API δέχεταιμέχρι 1 σελίδαδηλαδή 10 παιχνίδια. Θαβγάζειμήνυμαστον χρήστη να εισάγει μικρότερη ημερομηνία με χρήση αμυντικού προγραμματισμού  
if (SizeOfArrayWinningNumbers<= 10) {  
JsonArraycontentArray = jsonObj.getAsJsonArray("content");  
  
ArrayList<Integer>ArrayOfWinningNumber = newArrayList<>();  
ArrayList<Integer>ArrayOfJoker = newArrayList<>();  
  
//loop παιρνει τα δεδομενα από json τα bonus και τα προσθετει σε μια λιστα  
for (inti = 0; i<SizeOfArrayWinningNumbers; i++) {  
JsonObjectobj = (JsonObject) jsonObj.get("content").getAsJsonArray().get(i).getAsJsonObject().get("winningNumbers");  
JsonArrayarrayOfList = obj.get("list").getAsJsonArray();//pairneidedomenawinningnumbers  
for (intj = 0; j<arrayOfList.size(); j++) {  
ArrayOfWinningNumber.add(Integer.parseInt(arrayOfList.get(j).getAsString()));//taprostheteisthlista  
 }  
 }  
  
//loop παιρνει τα δεδομενα από json τα bonus και τα προσθετει σε μια λιστα  
for (inti = 0; i<SizeOfArrayBonusNumbers; i++) {  
JsonObjectobj = (JsonObject) jsonObj.get("content").getAsJsonArray().get(i).getAsJsonObject().get("winningNumbers");  
JsonArrayarrayOfBonus = obj.get("bonus").getAsJsonArray();//pairneidedomenaBONUSnumbers  
for (intj = 0; j<arrayOfBonus.size(); j++) {  
ArrayOfJoker.add(Integer.parseInt(arrayOfBonus.get(j).getAsString()));//ta prosthetei sth lista  
 }  
 }  
  
//ταξινόμισηλίστας  
 Collections.sort(ArrayOfWinningNumber);  
 Collections.sort(ArrayOfJoker);  
  
//μεταφοράσε string τωνσυχνάεμφανιζόμενωναριθμώνκαιμετατροπηωστεναμπεισε jTable  
int iteral = 0;  
for (int i = 1; i <= 45; i++) {  
int temp = Collections.frequency(ArrayOfWinningNumber, i);  
 countingNumbers[iteral] = Integer.toString(i);  
 ocurrs[iteral] = Integer.toString(temp);  
 delays[iteral] = "-";  
 iteral++;  
 }  
 iteral = 0;  
for (int i = 1; i <= 20; i++) {  
int temp = Collections.frequency(ArrayOfJoker, i);  
 bonusNumbersCoutningNumber[iteral] = Integer.toString(i);  
 bonusNumbersOcurrs[iteral] = Integer.toString(temp);  
 bonusNumbersDelays[iteral] = "-";  
 iteral++;  
 }  
//ΚαταχωρώστονπίνακατιςΚληρώσειςκαιτιςΚαθυστερήσεις  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray = new String[50];  
 String[] Column = new String[45];  
for (int i = 0; i < 45; i++) {  
 Column[i] = String.valueOf("");  
 emptyArray[i] = "";  
 }  
  
 DefaultTableModel model = new DefaultTableModel();  
 model.setColumnIdentifiers(Column);  
 model.addRow(countingNumbers);  
 model.addRow(emptyArray);  
 model.addRow(ocurrs);  
 model.addRow(emptyArray);  
 model.addRow(delays);  
 jTableWinningNumberStats.setModel(model);  
  
//Έξοδοςστοιχείωνστο jTableBonusWinningNumber  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray2 = new String[20];  
 String[] Column2 = new String[20];  
for (int i = 0; i < 20; i++) {  
 Column2[i] = String.valueOf("");  
 emptyArray2[i] = "";  
 }  
  
 DefaultTableModel model2 = new DefaultTableModel();  
 model2.setColumnIdentifiers(Column2);  
 model2.addRow(bonusNumbersCoutningNumber);  
 model2.addRow(emptyArray2);  
 model2.addRow(bonusNumbersOcurrs);  
 model2.addRow(emptyArray2);  
 model2.addRow(bonusNumbersDelays);  
 jTableWinningBonusNumberStats.setModel(model2);  
  
 jLabelDateFrom.setText(datePickerFrom.getText());  
 jLabelDateTo.setText(datePickerTo.getText());  
 jLabelTotalDraws.setText(Integer.toString(SizeOfArrayWinningNumbers));  
 } else {  
 JOptionPane.showMessageDialog(null, "Προβολήμέχρι 10 παιχνίδια. Παρακαλώ μικρύνετε το εύρος ημερομηνιών", "ΠΡΟΣΟΧΗ!!", JOptionPane.INFORMATION\_MESSAGE);  
 }  
 }  
 }

**privatevoid jButtonViewActionPerformed**(java.awt.event.ActionEvent evt) {   
 jTableWinningNumberStats.setVisible(true);  
 jTableWinningBonusNumberStats.setVisible(true);  
this.jButtonPdfPrint.setEnabled(true);  
 String urlToCall = "https://api.opap.gr/games/v1.0/5104/statistics";  
 OkHttpClient client = new OkHttpClient();  
 String responseString = null;  
 Request request = new Request.Builder().url(urlToCall).build();  
  
int iteral = 0; //for counter use into for..each  
  
try (Response response = client.newCall(request).execute()) {  
if (response.isSuccessful() && response.body() != null) {  
 responseString = response.body().string();  
 System.out.println(responseString);  
 }  
  
 GsonBuilder builder = new GsonBuilder();  
 builder.setPrettyPrinting();  
 Gson gson = builder.create();  
  
 JsonObject jsonObj = gson.fromJson(responseString, JsonObject.class);  
  
 JsonObject StatisticsObject = jsonObj.getAsJsonObject("header"); //winningNumber Object  
 JsonArray winningNumbers = jsonObj.get("numbers").getAsJsonArray();  
 JsonArray statsbonusNumbers = jsonObj.get("bonusNumbers").getAsJsonArray();  
  
for (int i = 0; i < 3; i++) {  
 header[0] = StatisticsObject.get("dateFrom").getAsString();  
 header[1] = StatisticsObject.get("dateTo").getAsString();  
 header[2] = StatisticsObject.get("drawCount").getAsString();  
 }  
  
//έξοδοςστονπίνακα jTableWinningNumbers  
for (JsonElement winningNumbersArrayParsing : winningNumbers) {  
 JsonObject winningNumbersArrayStatistics = winningNumbersArrayParsing.getAsJsonObject();  
//ΚαταχωρώστονπίνακατιςΚληρώσειςτιςΚαθυστερήσειςκαιτουςαριθμού;  
 ocurrs[iteral] = winningNumbersArrayStatistics.get("occurrences").toString();  
 delays[iteral] = winningNumbersArrayStatistics.get("delays").toString();  
 countingNumbers[iteral] = winningNumbersArrayStatistics.get("number").toString();  
 iteral++;  
 }  
  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray = new String[50];  
 String[] Column = new String[45];  
for (int i = 0; i < 45; i++) {  
 Column[i] = String.valueOf("");  
 emptyArray[i] = "";  
 }  
  
 DefaultTableModel model = new DefaultTableModel();  
 model.setColumnIdentifiers(Column);  
 model.addRow(countingNumbers);  
 model.addRow(emptyArray);  
 model.addRow(ocurrs);  
 model.addRow(emptyArray);  
 model.addRow(delays);  
 jTableWinningNumberStats.setModel(model);  
  
//Έξοδοςστοιχείωνστο jTableBonusWinningNumber  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray2 = new String[20];  
 String[] Column2 = new String[20];  
for (int i = 0; i < 20; i++) {  
 Column2[i] = String.valueOf("");  
 emptyArray2[i] = "";  
 }  
 iteral = 0;  
for (JsonElement winningNumbersArrayParsing : statsbonusNumbers) {  
 JsonObject winningNumbersArrayStatistics = winningNumbersArrayParsing.getAsJsonObject();  
//ΚαταχωρώστονπίνακατιςΚληρώσειςκαιτιςΚαθυστερήσεις  
 bonusNumbersOcurrs[iteral] = winningNumbersArrayStatistics.get("occurrences").toString();  
 bonusNumbersDelays[iteral] = winningNumbersArrayStatistics.get("delays").toString();  
 bonusNumbersCoutningNumber[iteral] = winningNumbersArrayStatistics.get("number").toString();  
 iteral++;  
 }  
  
 DefaultTableModel model2 = new DefaultTableModel();  
 model2.setColumnIdentifiers(Column2);  
 model2.addRow(bonusNumbersCoutningNumber);  
 model2.addRow(emptyArray2);  
 model2.addRow(bonusNumbersOcurrs);  
 model2.addRow(emptyArray2);  
 model2.addRow(bonusNumbersDelays);  
 jTableWinningBonusNumberStats.setModel(model2);  
  
//Εξοδοςστα JLabel στατιστικών  
//epoh timestamp converter  
 Date dateFrom = new Date(Integer.parseInt(header[0]) \* 1000L);  
 SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
 String myDateFrom = format.format(dateFrom);  
  
//epoh timestamp converter  
 Date dateTo = new Date(Integer.parseInt(header[1]) \* 1000L);  
 String myDateTo = format.format(dateTo);  
  
 jLabelDateFrom.setText(myDateFrom);  
 jLabelDateTo.setText(myDateTo);  
 jLabelTotalDraws.setText(header[2]);  
  
 SimpleDateFormat format2 = new SimpleDateFormat("dd MMMM yyyy");  
 String myDateFrom2 = format2.format(dateFrom);  
 String myDateTo2 = format2.format(dateTo);  
 datePickerFrom.setText(myDateFrom2);  
 datePickerTo.setText(myDateTo2);  
  
 } catch (IOException e) {  
 System.out.println(e);  
 }  
 }   
  
**privatevoid jButtonPdfPrintActionPerformed**(java.awt.event.ActionEvent evt) {   
// Εκκίνησηπρογράμματοςεκτύπωσηςσε pdf  
  
try {  
 Document document = new Document(PageSize.A3.rotate());  
  
 OutputStream outputStream = new FileOutputStream(new File("StatisticsOutput.pdf"));  
  
 PdfWriter.getInstance(document, outputStream);  
  
 document.open();  
  
 PdfPTable NumberPDF\_tableWN = new PdfPTable(45);  
 PdfPCell[] NumberPDF\_cellWN = new PdfPCell[45];  
 document.add(new Paragraph("Welcome statistic output"));  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
  
 document.add(new Paragraph("Statistics for Winning Numbers"));  
 document.add(Chunk.NEWLINE); //adding empty new line  
//Κώδικαςεκτύπωσης array σε pdf αφορα WinningNumber  
//numbers  
 document.add(new Paragraph("Numbers: "));  
for (int i = 0; i < 45; i++) {  
 NumberPDF\_cellWN[i] = new PdfPCell(new Paragraph(countingNumbers[i]));  
 NumberPDF\_tableWN.addCell(NumberPDF\_cellWN[i]);  
 }  
 NumberPDF\_tableWN.setWidthPercentage(100);  
 document.add(NumberPDF\_tableWN);  
  
//Occurrences  
 document.add(new Paragraph("Occurrences: "));  
 PdfPTable occurrPDF\_tableWN = new PdfPTable(45);  
 PdfPCell[] occurrPDF\_cellWN = new PdfPCell[45];  
for (int i = 0; i < 45; i++) {  
 occurrPDF\_cellWN[i] = new PdfPCell(new Paragraph(ocurrs[i]));  
 occurrPDF\_tableWN.addCell(occurrPDF\_cellWN[i]);  
 }  
 occurrPDF\_tableWN.setWidthPercentage(100);  
 document.add(occurrPDF\_tableWN);  
  
//Delays  
 document.add(new Paragraph("Delays: "));  
 PdfPTable delaysPDF\_tableWN = new PdfPTable(45);  
 PdfPCell[] delaysPDF\_cellWN = new PdfPCell[45];  
for (int i = 0; i < 45; i++) {  
 delaysPDF\_cellWN[i] = new PdfPCell(new Paragraph(delays[i]));  
 delaysPDF\_tableWN.addCell(delaysPDF\_cellWN[i]);  
 }  
 delaysPDF\_tableWN.setWidthPercentage(100);  
 document.add(delaysPDF\_tableWN);  
  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
 document.add(Chunk.NEWLINE); //adding empty new line  
  
 document.add(new Paragraph("Statistics for Joker Winning Numbers"));  
 document.add(Chunk.NEWLINE); //adding empty new line  
//Κώδικαςεκτύπωσης array σε pdf αφορα Joker Winning Numbers  
//numbers  
 PdfPTable NumberPDF\_tableJN = new PdfPTable(45);  
 PdfPCell[] NumberPDF\_cellJN = new PdfPCell[45];  
 document.add(new Paragraph("Numbers: "));  
for (int i = 0; i < 45; i++) {  
 NumberPDF\_cellJN[i] = new PdfPCell(new Paragraph(countingNumbers[i]));  
 NumberPDF\_tableJN.addCell(NumberPDF\_cellJN[i]);  
 }  
 NumberPDF\_tableJN.setWidthPercentage(100);  
 document.add(NumberPDF\_tableJN);  
  
//Occurrences  
 document.add(new Paragraph("Occurrences: "));  
 PdfPTable occurrPDF\_tableJN = new PdfPTable(45);  
 PdfPCell[] occurrPDF\_cellJN = new PdfPCell[45];  
for (int i = 0; i < 45; i++) {  
 occurrPDF\_cellJN[i] = new PdfPCell(new Paragraph(ocurrs[i]));  
 occurrPDF\_tableJN.addCell(occurrPDF\_cellJN[i]);  
 }  
 occurrPDF\_tableJN.setWidthPercentage(100);  
 document.add(occurrPDF\_tableJN);  
  
//Delays  
 document.add(new Paragraph("Delays: "));  
 PdfPTable delaysPDF\_tableJN = new PdfPTable(45);  
 PdfPCell[] delaysPDF\_cellJN = new PdfPCell[45];  
for (int i = 0; i < 45; i++) {  
 delaysPDF\_cellJN[i] = new PdfPCell(new Paragraph(delays[i]));  
 delaysPDF\_tableJN.addCell(delaysPDF\_cellJN[i]);  
 }  
 delaysPDF\_tableJN.setWidthPercentage(100);  
 document.add(delaysPDF\_tableJN);  
  
 document.close();  
 outputStream.close();  
  
 File myFile = new File("StatisticsOutput.pdf");  
 Desktop.getDesktop().open(myFile);  
  
 } catch (DocumentException | IOException e) {  
 System.out.println(e);  
 }  
 }   
  
**privatevoid jButton3ActionPerformed**(java.awt.event.ActionEvent evt) {   
 datePickerFrom.setText("");  
 datePickerTo.setText("");  
  
 jLabelDateFrom.setText("");  
 jLabelDateTo.setText("");  
 jLabelTotalDraws.setText("");  
  
//Γίνεταιμηδενισμόςτωνπινάκωνώστεότανγίνεικαθαρισμόςφόρμαςναμηντυπώνειτύποταστο pdf  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray = new String[50];  
 String[] Column = new String[45];  
for (int i = 0; i < 45; i++) {  
 Column[i] = String.valueOf("");  
 emptyArray[i] = "";  
 }  
  
 DefaultTableModel model = new DefaultTableModel();  
 model.setColumnIdentifiers(Column);  
 model.addRow(emptyArray);  
 model.addRow(emptyArray);  
 model.addRow(emptyArray);  
 model.addRow(emptyArray);  
 model.addRow(emptyArray);  
 jTableWinningNumberStats.setModel(model);  
//Έξοδοςστοιχείωνστο jTableBonusWinningNumber  
//έξοδοςκενού column καισβήσιμοτηςκεφαλίδαςτουκάθε column γιαλόγουςαισθητικήςτουπρογράμματος  
 String[] emptyArray2 = new String[20];  
 String[] Column2 = new String[20];  
for (int i = 0; i < 20; i++) {  
 Column2[i] = String.valueOf("");  
 emptyArray2[i] = "";  
 }  
 DefaultTableModel model2 = new DefaultTableModel();  
 model2.setColumnIdentifiers(Column2);  
 model2.addRow(emptyArray2);  
 model2.addRow(emptyArray2);  
 model2.addRow(emptyArray2);  
 model2.addRow(emptyArray2);  
 model2.addRow(emptyArray2);  
 jTableWinningBonusNumberStats.setModel(model2);  
  
//αρχικοποίησηπινάκων visible καικουμπιούεκτύπωσης  
 jTableWinningNumberStats.setVisible(false);  
 jTableWinningBonusNumberStats.setVisible(false);  
 jButtonPdfPrint.setEnabled(false);  
 }

**GraphStats.java**

Στηνσυγκεκριμένηκλάσηπαρόλοπουστηναπαίτησηδεναναφέρεταικάτιγιαεμφάνισητωναποτελεσμάτωνσεγραφικήμορφή, θεωρήσαμεότικάτιτέτοιοπρέπειναεμφανίζεταιστονχρήστηαφού το πλήκτρο *«Προβολή στατιστικών στοιχείων κληρώσεων σε* ***γραφική μορφή****»* περιέχει την λέξη «γραφική μορφή».

Η κλάση παρουσιάζει τα παρακάτω στατιστικά στοιχεία για τα δεδομένα που είναι αποθηκευμένα στη βάση δεδομένων σε μορφή πινάκων και γραφημάτων αντίστοιχα:

• Συχνότητας εμφάνισης αριθμών (πέντε πιο συχνά εμφανιζόμενοι αριθμοί)

• Συχνότητα εμφάνισης αριθμών joker (πέντε πιο συχνά εμφανιζόμενοι αριθμοί)

• Μέσος όρος κερδών ανά κατηγορία

Επίσης δίνει στον χρήστη την δυνατότητα να επιλέξει ποια στατιστικά θα παραχθούν κάθε φορά καθώς και την δυνατότητα να επιλέξει ένα εύρος ημερομηνιών για το οποίο θα εμφανιστούν τα συγκεκριμένα στατιστικά (απαίτηση R5).

EntityManagerFactory emf = Persistence.createEntityManagerFactory("plh24\_ge3PU");  
 EntityManager em = emf.createEntityManager();  
 GameJpaController gjc = new GameJpaController(emf);  
  
/\*\*  
 \* Creates new form GraphStats  
 \*/  
**public GraphStats()**{  
 JFrame frame = new JFrame("Προβολή στατιστικών στοιχείων κληρώσεων σε γραφική μορφή");  
  
//Listener ώστε να ξεκινήσω την διαδικασία όταν το παράθυρο θα έχει ανοίξει  
//και να κλείσω το μενού επιλογών σε περίπτωση που ο χρήστης κλείσει το παράθυρο  
this.addWindowListener(new WindowAdapter() {  
// Invoked when a window has been opened.  
 @Override  
publicvoid windowOpened(WindowEvent e) {  
//Listener ώστε να κλείσω το παράθυρο σε περίπτωση που ο χρήστης κλείσει το μενού επιλογών  
 frame.addWindowListener(new java.awt.event.WindowAdapter() {  
 @Override  
publicvoid windowClosing(java.awt.event.WindowEvent windowEvent) {  
 dispose();  
 }  
 });  
 frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 JPanel panel = new JPanel();  
 panel.setLayout(null);  
  
//Δημιουργία CheckBoxes  
 JCheckBox checkbox1 = new JCheckBox("Συχνότητας εμφάνισης αριθμών");  
 Dimension size = checkbox1.getPreferredSize();  
 checkbox1.setBounds(10, 10, size.width, size.height);  
panel.add(checkbox1);  
  
JCheckBoxcheckbox2 = newJCheckBox("Συχνότηταεμφάνισηςαριθμώνjoker");  
Dimensionsize25 = checkbox2.getPreferredSize();  
checkbox2.setBounds(10, 30, size25.width, size25.height);  
panel.add(checkbox2);  
  
JCheckBoxcheckbox3 = newJCheckBox("Μέσοςόροςκερδώνανάκατηγορία");  
Dimensionsize26 = checkbox3.getPreferredSize();  
checkbox3.setBounds(10, 50, size26.width, size26.height);  
panel.add(checkbox3);  
  
//ΔημιουργίαLabel  
JLabellblFrom = newJLabel();  
lblFrom.setText("Από:");  
Dimensionsize30 = lblFrom.getPreferredSize();  
lblFrom.setBounds(15, 100, size30.width, size30.height);  
panel.add(lblFrom);  
  
// ΔημιουργίαDatePickerfrom  
DatePickerdatePickerFrom = newDatePicker();  
DatePickerSettingsdateSettings = newDatePickerSettings();  
Dimensionsize1 = datePickerFrom.getPreferredSize();  
datePickerFrom.setBounds(45, 97, size1.width, size1.height);  
dateSettings.setFormatForDatesCommonEra("dd/MM/yyyy");  
datePickerFrom.setDateToToday();  
datePickerFrom.setSettings(dateSettings);  
datePickerFrom.setEnabled(false);  
//Προσθήκηστοpanelτουframe  
panel.add(datePickerFrom);  
  
//Δημιουργία Label  
 JLabel lblTo = new JLabel();  
 lblTo.setText("Έως:");  
 Dimension size0 = lblTo.getPreferredSize();  
 lblTo.setBounds(215, 100, size0.width, size0.height);  
 panel.add(lblTo);  
  
// Δημιουργία DatePicker to  
 DatePicker datePickerTo = new DatePicker();  
 DatePickerSettings dateSettings2 = new DatePickerSettings();  
 Dimension size2 = datePickerTo.getPreferredSize();  
 datePickerTo.setBounds(247, 97, size2.width, size2.height);  
 dateSettings2.setFormatForDatesCommonEra("dd/MM/yyyy");  
 datePickerTo.setDateToToday();  
 datePickerTo.setSettings(dateSettings2);  
 datePickerTo.setEnabled(false);  
//Προσθήκηστο panel του frame  
 panel.add(datePickerTo);  
  
 JCheckBox checkbox4 = new JCheckBox("Εντόςημερομηνιών:");  
 Dimension size27 = checkbox4.getPreferredSize();  
 checkbox4.setBounds(10, 70, size27.width, size27.height);  
 checkbox4.addItemListener((ItemEvent e1) -> {  
if (e1.getStateChange() == 2) {  
 datePickerFrom.setEnabled(false);  
 datePickerTo.setEnabled(false);  
 } else {  
 datePickerFrom.setEnabled(true);  
 datePickerTo.setEnabled(true);  
 }  
 });  
 panel.add(checkbox4);  
  
// Δημιουργία Button  
 JButton button = new JButton("ΠροβολήΣτατιστικών");  
 Dimension size3 = button.getPreferredSize();  
 button.setBounds(130, 140, size3.width, size3.height);  
  
//ActionListener γιατο Button  
 button.addActionListener((ActionEvent ev) -> {  
  
//Συχνότηταεμφάνισηςαριθμών  
if (checkbox1.isSelected()) {  
try {  
 numbersFrequency(datePickerFrom, datePickerTo, checkbox4.isSelected(), frame);  
 createGraph(tblNums, panelNums, "Συχνότηταεμφάνισηςαριθμών", "Αριθμοί", "Εμφανίσεις");  
 } catch (ParseException ex) {  
 Logger.getLogger(GraphStats.class.getName()).log(Level.SEVERE, null, ex);  
 }  
 } else {  
 jPanel2.setVisible(false);  
 }  
  
//Συχνότηταεμφάνισηςαριθμών joker  
if (checkbox2.isSelected()) {  
try {  
 jokerFrequency(datePickerFrom, datePickerTo, checkbox4.isSelected(), frame);  
 createGraph(tblJoker, panelJoker, "Συχνότηταεμφάνισηςαριθμών Joker", "Αριθμοί Joker", "Εμφανίσεις");  
 } catch (ParseException ex) {  
 Logger.getLogger(GraphStats.class.getName()).log(Level.SEVERE, null, ex);  
 }  
 } else {  
 jPanel1.setVisible(false);  
 }  
  
//ΜέσοςόροςΚερδώναναΚατηγορία  
if (checkbox3.isSelected()) {  
try {  
 avgPerCategory(datePickerFrom, datePickerTo, checkbox4.isSelected(), frame);  
 createGraph(tblCat, panelCat, "Μέσοςόροςκερδώνανάκατηγορία", "ΚατηγορίαΚερδών", "ΜέσοςΌροςΚερδών");  
 } catch (ParseException ex) {  
 Logger.getLogger(GraphStats.class.getName()).log(Level.SEVERE, null, ex);  
 }  
 } else {  
 jPanel3.setVisible(false);  
 }  
 });  
//Προσθήκηστο panel του frame  
 panel.add(button);  
  
 frame.getContentPane().add(panel, BorderLayout.CENTER);  
 frame.setSize(430, 230);  
 frame.setResizable(false);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }  
  
//Κλείσιμο GraphStats  
 @Override  
publicvoid windowClosing(WindowEvent e) {  
 frame.dispose();  
 }  
 });  
  
this.setTitle("Προβολήστατιστικώνστοιχείωνκληρώσεωνσεγραφικήμορφή");//Τίτλοςτηςφόρμας  
initComponents();  
 }  
  
**privatevoidresizeColumnWidth**(JTabletable) {  
finalTableColumnModelcolumnModel = table.getColumnModel();  
for (intcolumn = 0; column<table.getColumnCount(); column++) {  
intwidth = 15; // Minwidth  
for (introw = 0; row<table.getRowCount(); row++) {  
TableCellRendererrenderer = table.getCellRenderer(row, column);  
Componentcomp = table.prepareRenderer(renderer, row, column);  
width = Math.max(comp.getPreferredSize().width + 1, width);  
 }  
if (width> 300) {  
width = 300;  
 }  
columnModel.getColumn(column).setPreferredWidth(width);  
 }  
 }  
  
**privatevoidavgPerCategory**(DatePickerdatePickerFrom, DatePickerdatePickerTo, booleandates, JFrameframe) {  
try {  
//λίστααντικειμένωνgames  
List<Game>games = newArrayList<>();  
intc = 0; //Μετρητήςγιατοπλήθοςτωνκληρώσεων  
//Αθροίσματαγιατακέρδηκάθεκατηγορίας  
doublesum5\_1 = 0;  
doublesum5 = 0;  
doublesum4\_1 = 0;  
doublesum4 = 0;  
doublesum3\_1 = 0;  
doublesum3 = 0;  
doublesum2\_1 = 0;  
doublesum1\_1 = 0;  
  
if (dates) {  
DateFormatformat = newSimpleDateFormat("yyyy-MM-dd");  
java.util.DatedateFrom = format.parse(datePickerFrom.getDateStringOrEmptyString());  
java.util.DatedateTo = format.parse(datePickerTo.getDateStringOrEmptyString());  
  
//ΒρίσκωταdrawIdsτωνκληρώσεων, μεταξύτωνημερομηνιώνπουέδωσεοχρήστης,  
//καιτααποθηκεύωστηνλίστααντικειμένωνgames  
em.getTransaction().begin();  
Queryquery = em.createNamedQuery("Game.findBetweenDrawdate");  
query.setParameter("drawdateFrom", dateFrom);  
query.setParameter("drawdateTo", dateTo);  
  
//λίστααντικειμένωνgames  
games = query.getResultList();  
if (games.isEmpty()) {  
JOptionPane.showMessageDialog(null, "Δενυπάρχουνδιαθέσιμαδεδομένα", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
em.getTransaction().commit();  
return;  
 }  
  
Queryq = em.createNativeQuery("SELECT \* FROMPRIZECATEGORIESaWHEREa.DRAWIDBETWEEN ?1 AND ?2");  
q.setParameter(1, games.get(0).getDrawid());  
q.setParameter(2, games.get(games.size() - 1).getDrawid());  
List<Object>result = (List<Object>) q.getResultList();  
em.getTransaction().commit();  
  
c = 0;  
Iteratoritr = result.iterator();  
while (itr.hasNext()) {  
Object[] obj = (Object[]) itr.next();  
  
int prizeId = Integer.parseInt(String.valueOf(obj[2]));  
double divident = (double) (obj[3]);  
int winners = Integer.parseInt(String.valueOf(obj[4]));  
  
switch (prizeId) {  
case 1:  
 sum5\_1 += divident \* winners;  
 c += 1;  
break;  
case 2:  
 sum5 += divident \* winners;  
break;  
case 3:  
 sum4\_1 += divident \* winners;  
break;  
case 4:  
 sum4 += divident \* winners;  
break;  
case 5:  
 sum3\_1 += divident \* winners;  
break;  
case 6:  
 sum3 += divident \* winners;  
break;  
case 7:  
 sum2\_1 += divident \* winners;  
break;  
case 8:  
 sum1\_1 += divident \* winners;  
break;  
 }  
 }  
 } else {  
 List<Prizecategories> prizeCategories = new ArrayList<>();  
//Select όλωντωνεγγραφώντουπίνακα PrizeCategories  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Prizecategories.findAll");  
  
 prizeCategories = query.getResultList();  
 em.getTransaction().commit();  
  
 c = 0;  
//γιακάθεκλήρωσηβρίσκωτοάθροισματωνκερδώνγιακάθεκατηγορία, καθώςκαιτοπλήθοςτωνκληρώσεων  
for (Prizecategories category : prizeCategories) {  
switch (category.getPrizeid()) {  
case 1:  
 sum5\_1 += category.getDivedent() \* category.getWinners();  
 c += 1;  
break;  
case 2:  
 sum5 += category.getDivedent() \* category.getWinners();  
break;  
case 3:  
 sum4\_1 += category.getDivedent() \* category.getWinners();  
break;  
case 4:  
 sum4 += category.getDivedent() \* category.getWinners();  
break;  
case 5:  
 sum3\_1 += category.getDivedent() \* category.getWinners();  
break;  
case 6:  
 sum3 += category.getDivedent() \* category.getWinners();  
break;  
case 7:  
 sum2\_1 += category.getDivedent() \* category.getWinners();  
break;  
case 8:  
 sum1\_1 += category.getDivedent() \* category.getWinners();  
break;  
 }  
 }  
 }  
  
//Πίνακαςγιατουςμέσουςόρουςκερδών  
 String[] row = {"Μέσοςόροςκερδών:", String.format(Locale.getDefault(), "%,.2f", (sum5\_1 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum5 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum4\_1 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum4 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum3\_1 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum3 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum2\_1 / c)),  
 String.format(Locale.getDefault(), "%,.2f", (sum1\_1 / c))};  
  
//Πέρασμαστο Jtable  
 DefaultTableModel dTable1 = (DefaultTableModel) tblCat.getModel();  
 dTable1.addRow(row);  
  
 resizeColumnWidth(tblCat);//Προσαρμογήτουπλάτουςτωνστηλώντου Jtable σταπεριεχόμενατωνστηλών  
 frame.dispose();  
  
 } catch (HeadlessException | ParseException ex) {  
 JOptionPane.showMessageDialog(null, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 System.out.println(ex);  
 }  
 }  
  
**privatevoid numbersFrequency**(DatePicker datePickerFrom, DatePicker datePickerTo, boolean dates, JFrame frame) {  
int[][] numbers = newint[45][2];  
for (int i = 0; i < numbers.length; i++) {  
 numbers[i][0] = i + 1;  
 numbers[i][1] = 0;  
 }  
try {  
//λίστααντικειμένων games  
 List<Game> games = new ArrayList<>();  
if (dates) {  
 DateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
 java.util.Date dateFrom = format.parse(datePickerFrom.getDateStringOrEmptyString());  
 java.util.Date dateTo = format.parse(datePickerTo.getDateStringOrEmptyString());  
  
//Βρίσκωτα drawIds τωνκληρώσεων, μεταξύτωνημερομηνιώνπουέδωσεοχρήστης,  
//καιτααποθηκεύωστηνλίστααντικειμένων games  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findBetweenDrawdate");  
 query.setParameter("drawdateFrom", dateFrom);  
 query.setParameter("drawdateTo", dateTo);  
  
//λίστααντικειμένων games   
 games = query.getResultList();  
 em.getTransaction().commit();  
 } else {  
//Βρίσκωτα drawIds τωνκληρώσεωνκαιτααποθηκεύωστηνλίστααντικειμένων games  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findAll");  
  
 games = query.getResultList();  
 em.getTransaction().commit();  
 }  
if (games.isEmpty()) {  
 JOptionPane.showMessageDialog(null, "Δενυπάρχουνδιαθέσιμαδεδομένα", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
return;  
 }  
  
//Γιακάθεκλήρωσηπουβρέθηκεαποθηκεύωτοναριθμό Joker στονπίνακασυχνοτήτων jokers  
for (Game game : games) {  
 numbers[game.getNum1() - 1][0] = game.getNum1();  
 numbers[game.getNum1() - 1][1] += 1;  
  
 numbers[game.getNum2() - 1][0] = game.getNum2();  
 numbers[game.getNum2() - 1][1] += 1;  
  
 numbers[game.getNum3() - 1][0] = game.getNum3();  
 numbers[game.getNum3() - 1][1] += 1;  
  
 numbers[game.getNum4() - 1][0] = game.getNum4();  
 numbers[game.getNum4() - 1][1] += 1;  
  
 numbers[game.getNum5() - 1][0] = game.getNum5();  
 numbers[game.getNum5() - 1][1] += 1;  
 }  
  
//Ταξινόμησηπίνακα jokers μεβάσητηνσυχότηταεμφάνισης  
 Arrays.sort(numbers, (int[] o1, int[] o2) -> Integer.compare(o2[1], o1[1]));  
  
//Εμφάνισηαποτελεσμάτωνστο Jtable1  
 DefaultTableModel dTable1 = (DefaultTableModel) tblNums.getModel();  
 String[] row = new String[6];  
 row[0] = "Εμφανίσεις:";  
for (int i = 0; i < 5; i++) {  
 JTableHeader th = tblNums.getTableHeader();  
 TableColumnModel tcm = th.getColumnModel();  
 TableColumn tc = tcm.getColumn(i + 1);  
 tc.setHeaderValue(numbers[i][0]);  
 th.repaint();  
  
 row[i + 1] = String.valueOf(numbers[i][1]);  
 }  
 dTable1.addRow(row);  
 resizeColumnWidth(tblNums);  
 frame.dispose();  
  
 } catch (HeadlessException | ParseException ex) {  
 JOptionPane.showMessageDialog(null, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 System.out.println(ex);  
 }  
 }  
  
**privatevoid jokerFrequency**(DatePicker datePickerFrom, DatePicker datePickerTo, boolean dates, JFrame frame) {  
int[][] jokers = newint[20][2];  
for (int i = 0; i < jokers.length; i++) {  
 jokers[i][0] = i + 1;  
 jokers[i][1] = 0;  
 }  
try {  
//λίστααντικειμένων games  
 List<Game> games = new ArrayList<>();  
if (dates) {  
 DateFormat format = new SimpleDateFormat("yyyy-MM-dd");  
 java.util.Date dateFrom = format.parse(datePickerFrom.getDateStringOrEmptyString());  
 java.util.Date dateTo = format.parse(datePickerTo.getDateStringOrEmptyString());  
  
//Βρίσκωτα drawIds τωνκληρώσεων, μεταξύτωνημερομηνιώνπουέδωσεοχρήστης,  
//καιτααποθηκεύωστηνλίστααντικειμένων games  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findBetweenDrawdate");  
 query.setParameter("drawdateFrom", dateFrom);  
 query.setParameter("drawdateTo", dateTo);  
  
//λίστααντικειμένων games   
 games = query.getResultList();  
 em.getTransaction().commit();  
 } else {  
//Βρίσκωτα drawIds τωνκληρώσεωνκαιτααποθηκεύωστηνλίστααντικειμένων games  
 em.getTransaction().begin();  
 Query query = em.createNamedQuery("Game.findAll");  
  
 games = query.getResultList();  
 em.getTransaction().commit();  
 }  
  
if (games.isEmpty()) {  
 JOptionPane.showMessageDialog(null, "Δενυπάρχουνδιαθέσιμαδεδομένα", this.getTitle(), JOptionPane.WARNING\_MESSAGE);  
return;  
 }  
  
//Γιακάθεκλήρωσηπουβρέθηκεαποθηκεύωτοναριθμό Joker στονπίνακασυχνοτήτων jokers  
for (Game game : games) {  
 jokers[game.getBonus() - 1][0] = game.getBonus();  
 jokers[game.getBonus() - 1][1] += 1;  
 }  
  
//Ταξινόμησηπίνακα jokers μεβάσητηνσυχότηταεμφάνισης  
 Arrays.sort(jokers, (int[] o1, int[] o2) -> Integer.compare(o2[1], o1[1]));  
  
//Εμφάνισηαποτελεσμάτωνστο Jtable1  
 DefaultTableModel dTable1 = (DefaultTableModel) tblJoker.getModel();  
 String[] row = new String[6];  
 row[0] = "Εμφανίσεις:";  
for (int i = 0; i < 5; i++) {  
JTableHeader th = tblJoker.getTableHeader();  
 TableColumnModel tcm = th.getColumnModel();  
 TableColumn tc = tcm.getColumn(i + 1);  
 tc.setHeaderValue(jokers[i][0]);  
 th.repaint();  
  
 row[i + 1] = String.valueOf(jokers[i][1]);  
 }  
 dTable1.addRow(row);  
 resizeColumnWidth(tblJoker);  
 frame.dispose();  
 } catch (HeadlessException | ParseException ex) {  
 JOptionPane.showMessageDialog(null, ex, "Error", JOptionPane.ERROR\_MESSAGE);  
 System.out.println(ex);  
 }  
 }  
  
//Μέθοδοςγιατηνδημιουργίαγραφήματος  
**privatevoid createGraph**(JTable table, JPanel panel, String title, String xLabel, String yLabel) throws ParseException {  
 DefaultCategoryDataset dataset = new DefaultCategoryDataset();  
 DefaultTableModel dtm = (DefaultTableModel) table.getModel();  
int nCol = dtm.getColumnCount();  
for (int i = 1; i < nCol; i++) {  
 JTableHeader th = table.getTableHeader();  
 TableColumnModel tcm = th.getColumnModel();  
 TableColumn tc = tcm.getColumn(i);  
  
if ("Μέσοςόροςκερδώνανάκατηγορία".equals(title)) {  
 DecimalFormatSymbols symbols = new DecimalFormatSymbols();  
 symbols.setDecimalSeparator('.');  
 DecimalFormat format = new DecimalFormat("0.#", DecimalFormatSymbols.getInstance(Locale.getDefault()));  
 format.setDecimalFormatSymbols(symbols);  
float f = format.parse(dtm.getValueAt(0, i).toString()).floatValue();  
  
 dataset.setValue(f, xLabel, tc.getHeaderValue().toString());  
 } else {  
 dataset.setValue(Integer.parseInt(dtm.getValueAt(0, i).toString()), xLabel, tc.getHeaderValue().toString());  
 }  
 }  
  
 JFreeChart chart = ChartFactory.createBarChart(title, xLabel, yLabel, dataset, PlotOrientation.VERTICAL, false, true, false);  
 CategoryPlot p = chart.getCategoryPlot();  
 p.setRangeGridlinePaint(Color.BLACK);  
  
 ChartPanel pa = new ChartPanel(chart);  
 panel.removeAll();  
 panel.add(pa, BorderLayout.CENTER);  
 panel.validate();  
 }

## ΕρώτημαΔ – ΣυνολικόςΈλεγχοςκαιΕκτέλεσητηςΕφαρμογής

*[Να εκτελέσετε την εφαρμογή και να εισαγάγετε σε μορφή πίνακα εικόνες (screendumps), όπου θα φαίνεται το αποτέλεσμα της εκτέλεσης.Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ. Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.*

*Εάν δεν έχετε υλοποιήσει μια απαίτηση γράψτε στην αντίστοιχη γραμμή ΔΕΝ ΥΛΟΠΟΙΗΘΗΚΕ.*

*Εάν δεν έχετε υλοποιήσει ΠΛΗΡΩΣ μια απαίτηση γράψτε στην αντίστοιχη γραμμή ΔΕΝ ΥΛΟΠΟΙΗΘΗΚΕ ΠΛΗΡΩΣ, προσδιορίζοντας στη συνέχεια το βαθμό υλοποίησής της.]*

|  |  |
| --- | --- |
| **Απαίτηση** | **Εικόνα** |
| R1 | Ο χρήστης έχει την δυνατότητα να επιλέξει για ποιο παιχνίδι του ΟΠΑΠ θέλει να δει και να επεξεργαστεί δεδομένα. Στην παρούσα φάση έχει γίνει υλοποίηση μόνο για το παιχνίδι ΤΖΟΚΕΡ.      Κάνοντας κλικ στο button«4. Έξοδος» εμφανίζεται το παρακάτω: |
| R2 | Κάνοντας κλικ στο button«1. Διαχείριση δεδομένων ΤΖΟΚΕΡ» εμφανίζεται το παρακάτω παράθυρο:    Ο χρήστης σε αυτό το παράθυρο μπορεί να επιλέξει το idμιας και να προβληθούν λεπτομέρειες για την εν λόγω κλήρωση:    Επίσης μπορεί να επιλέξει εύρος ημερομηνιών, στην συνέχεια να επιλέξει από λίστα με τα id’sτων κληρώσεων που έχουν γίνει εντός τους εύρους ημερομηνιών το idτης κλήρωσης που επιθυμεί και στην συνέχεια να προβληθούν τα δεδομένα για την συγκριμένη κλήρωση:        Κάνοντας κλικ στο button«Αποθήκευση στην βάση δεδομένων», ο χρήστης μπορεί να αποθηκεύσει τις λεπτομέρειες της κλήρωσης. Μετά την αποθήκευση εμφανίζεται μήνυμα επιτυχίας:    Σε περίπτωση που η κλήρωση που θέλει να αποθηκεύσει ο χρήστης υπάρχει ήδη στην βάση δεδομένων, η αποθήκευση δεν πραγματοποιείται και εμφανίζεται κατάλληλο μήνυμα:  Κάνοντας κλικ στο button«Διαγραφή δεδομένων παιχνιδιού», εμφανίζεται λίστα με τα τις αποθηκευμένες, στην βάση δεδομένων κληρώσεις, και χρήστης μπορεί να επιλέξει το idτην κλήρωσης που θέλει να διαγραφεί.  Πριν την διαγραφή εμφανίζεται στον χρήστη μήνυμα επιβεβαίωσης:  Κάνοντας κλικ στο button«Διαγραφή δεδομένων παιχνιδιούεντός εύρους ημερομηνιών», εμφανίζεται παράθυρο οπού ο χρήστης επιλέγει εύρος ημερομηνιών για την διαγραφή των κληρώσεων που υπάρχουν εντός του εύρους:  Όπως και πριν έτσι και εδώ εμφανίζεται μήνυμα επιβεβαίωσης πριν την διαγραφή: |
| R3 | Κάνοντας κλικ στο button«2. Προβολή δεδομένων ΤΖΟΚΕΡ ανά μήνα για συγκεκριμένο έτος», από την αρχική οθόνη, εμφανίζεται παράθυρο στον χρήστη οπού μπορεί να επιλέξει το έτος και τον μήνα του έτους για τον οποίον υπάρχουν αποθηκευμένα δεδομένα στην βάση δεδομένων:  Επιλέγοντας έτος και μήνα, εμφανίζεται στον χρήστη τοπαρακάτω παράθυρο με τα δεδομένα: |
| R4 | Κάνοντας κλικ στο button«Προβολή στατιστικών δεδομένων ΤΖΟΚΕΡ και εκτύπωση σε αρχείο pdf», από την αρχική οθόνη,εμφανίζεται το παρακάτω παράθυρο στον χρήστη, όπου κάνοντας κλικ στο button«OverallΠροβολή» εμφανίζονται τα στατιστικά των αριθμών:  Επίσης ο χρήστης έχει την δυνατότητα να επιλέξει εύρος ημερομηνιών κάνοντας κλικ στο button«Προβολή» εμφανίζονται τα στατιστικά των αριθμών για το δοθέν εύρος ημερομηνιών, όπως φαίνεται παρακάτω:  Κάνοντας κλικ στο button«Εκτύπωση» αποθηκεύονται τα στατιστικά των αριθμών σε αρχείο pdf. *(Το αρχείο αποθηκεύεται στον φάκελο του project)*  Κάνοντας κλικ στο button«Καθαρισμός φόρμας», το παράθυρο επανέρχεται στην αρχική του μορφή και καθαρίζονται όλα τα πεδία.  Κάνοντας κλικ στο button«Προβολή στατιστικών στοιχείων κληρώσεων σε γραφική μορφή», ανοίγει παράθυρο για την απαίτηση R5 και παρουσιάζεται παρακάτω. |
| R5 | Σε αυτό το παράθυρο ο χρήστης καλείται να επιλέξει ποια στατιστικά θέλει να παραχθούν και αν θέλει να επιλέξει εύρος ημερομηνιών για το οποίο θα εμφανιστούν τα επιλεγμένα στατιστικά. Τα στατιστικά παράγονται με βάση τις κληρώσεις που είναι αποθηκευμένες στην βάση δεδομένων.  Με βάση τις παραπάνω επιλογές εμφανίζεται το παρακάτω παράθυρο, όπου φαίνονται τα στατιστικά σε μορφή πινάκων καθώς και σε γραφική μορφή.  Επιλογή διαφορετικών στατιστικών που θα παραχθούν εντός εύρους ημερομηνιών:  Τα στατιστικά που παράγωνται βάση των παραπάνω επιλογών: |

# ΚΡΙΤΙΚΟΣΑΠΟΛΟΓΙΣΜΟΣΤΟΥΕΡΓΟΥ

*[Κάντεέναναναλυτικό και κριτικό απολογισμό του έργου αναφέροντας*

#### τα προβλήματα που αντιμετωπίσατε, τι θα αλλάζατε στο έργο αν είχατε τη δυνατότητα,

#### τις αποκλίσεις που είχατε στις επαναλήψεις σε σχέση με αυτά που ορίζονται στην εργασία,

#### το χρόνο που απαιτήθηκε για την κάθε δραστηριότητα με βάση τα στοιχεία που κρατήσατε και σε σχέση με το χρόνο που προϋπολογίσατε,

#### πως αξιολογείτε το εργαλείο συνεργασίας trello και με ποιο τρόπο σας βοήθησε στην καθημερινή σας εργασία

#### το πιο πολύτιμο μέλος της ομάδας σας και τον τρόπο που το επιλέξατε,

#### τους κινδύνους που πραγματώθηκαν, καθώς και πώς αποκριθήκατε σε αυτούς,

#### τι θα αλλάζατε σε ένα επόμενο αντίστοιχο έργο, καθώς και

#### ποια ήταν τα θετικά σημεία που αποκομίσατε από αυτή την εργασία].

1. Λόγο απειρίας από την πλειοψηφία των μελών η ομάδα έπρεπε να αφιερώσει αρκετό χρόνο στην αναζήτηση γνώσης για να φέρει σε πέρας το ζητούμενο έργο. Επίσης κάποια από τα μέλη είχαν χρόνια να ασχοληθούν με την γλώσσα SQLκαι έπρεπε να αφιερωθεί κάποιος χρόνος σε αυτήν. Το μέγεθος της εργασίας,λόγο ομαδικής και όχι ατομικής, αποτελεί από μόνο του μία πρόκληση, λόγο της διευρυμένης περιόδου ενασχόλησης με αυτήν, στον οικογενειακό χρονοπρογραμματισμό.

Όσον αφορά αλλαγών που θα κάναμε στο έργο θα ήταν ίσος η διάθεση χρόνου για την εξοικείωση με νέες διαθέσιμες διαδικτυακές εφαρμογές και χρησιμοποίηση αυτών ώστε η πρόοδος του κάθε μέλους πάνω στο κομμάτι του κώδικα και οι αλλαγές που ενδεχομένως να κάνει ένα μέλος να είναι διαθέσιμες σεrealtimeπροβολή από τα υπόλοιπα μέλη της ομάδας κάτι που θα επιτάχυνε σε μεγάλο βαθμό την συγγραφή του συνόλου του κώδικα.

1. Όσων αφορά τυχών αποκλίσεωνδεν παρατηρήσαμε κάποια σημαντική απόκλιση από τα αρχικά σχέδια καθώς, λόγο απειρίας αλλά και ομόφωνης τοποθέτησης στο να δουλέψουμε με όσο το δυνατών λιγότερο άγχος με κατανόηση στις προσωπικές ιδιαιτέρες ανάγκες του κάθε μέλους και με στόχο να πετύχουμε το καλύτερο δυνατό αποτέλεσμα, τα χρονικά περιθώρια ήταν προσαρμοσμένα σε αυτήν την λογική. Ενδεχομένως μία πιο έμπειρη ομάδα να έβαζε ποιο ασφυκτικά περιθώρια και εν τέλη να τελείωνε το έργο σε ποιο σύντομο χρόνο.
2. Ο παρακάτω πίνακας δείχνει τους εκτιμώμενους χρόνους ολοκλήρωσης και τους πραγματικούς χρόνους ολοκλήρωσης ανά δραστηριότητα.



1. Το Trelloαποτέλεσε σημαντικό σύμμαχο στο όλο εγχείρημα καθώς η ευκολία τόσο στην εκμάθηση όσο και στον χειρισμό του μας βοήθησε στο να οργανώσουμε τις δραστηριότητες χωρίς να σπαταλήσουμε χρόνο με κάποια πολύπλοκη και δύσχρηστη εφαρμογή. Επίσης ήταν ένας εύκολος τρόπος να ανατρέχουμε ώστε να βάζουμε τα όρια στον προς υλοποίηση κώδικά μας τόσο ανάμεσα στα Sprintόσο και ανάμεσα στα μέλη της ομάδας.
2. Πολυτιμότερο μέλος της ομάδας αποτέλεσε ο Πιτυανός Εμμανουήλ και ScrumMasterτης ομάδας καθώς με τις γνώσεις και την διάθεση του να βοηθήσει τα υπόλοιπα μέλη της ομάδας αποτέλεσε σημαντικό αγχολυτικό παράγοντα για την ομάδα. Η εμπειρία του ήταν καθοριστική στον διαχωρισμό και οργάνωση των δραστηριοτήτων έτσι ώστε να μην επιβαρυνθεί κάποιο μέλος με υπερβολικό φόρτο εργασίας.

α) Ο Αρχικός κίνδυνος που έπρεπε να αντιμετωπίσουμε προερχόταν από το ότι τα μέλη της ομάδας δεν γνωρίζονταν μεταξύ τους με αποτέλεσμα να μην γνωρίζουμε ο ένας τα θέλω του άλλου και τι προσδοκίες είχε από την εργασία αυτή, για το αν θα μπορούσε η ομάδα να δέσει και να βγάλει σε πέρας το έργο ή θα δημιουργούνταν αψιμαχίες δυσαρέσκειες και εν τέλει αποτυχία της ομάδας. Από την πρώτη κιόλας επικοινωνία έγινε εμφανής η διάθεση όλων τόσο στο να επικοινωνήσουν με τα άλλα μέλη της ομάδας όσο και να σεβαστούν τις οικογενειακές και επαγγελματικές υποχρεώσεις του κάθε μέλους δείχνοντας ευελιξία σε όλους τους τομείς που σχετίζονταν με το έργο (πχ. ώρες επικοινωνίας, επιλογή δραστηριοτήτων κτλ.).

β) Η έλλειψη ανάλογης εμπειρίας από την πληθώρα των μελών της ομάδας αποτέλεσε παράγοντα άγχους που όμως λύθηκε από την καθοριστική παρέμβαση του ScrumMaster.

7) Νομίζω ότι εάν έπρεπε να ασχοληθούμε με ένα παρόμοιο έργο θα μπορούσαμε να χρησιμοποιήσουμε περισσότερες διαθέσιμες εφαρμογές όπως πχ το GitHubώστε το χτίσιμο του κώδικα από τα μέλη να είναι αμεσότερο μειώνοντας και το χρόνο που χρειάζεται για την συνένωση των κομματιών του κώδικα από τα μέλη.

8) Η εμπειρία του να δουλεύεις με μία ομάδα ανθρώπων ήταν για τα περισσότερα μέλη πρωτόγνωρη. Το αίσθημα ευθύνης απέναντι στα υπόλοιπα μέλη. Η πληθώρα εφαρμογών και γνώσεων που χρειάζεται ακόμα και ένα μικρής κλίμακας έργο. Τέλος το να γνωρίζεις άλλους δύο αξιόλογους ανθρώπους.

# ΑΝΑΦΟΡΕΣ

*[Παραθέστε τις βιβλιογραφικές αναφορές (βιβλία, άρθρα, σύνδεσμοι στο διαδίκτυο) που χρησιμοποιήσατε για την υλοποίηση της εργασίας.]*

* Βεσκούκης Β. (2008), *Σχεδιασμός Λογισμικού Τόμος Β΄ - Τεχνολογία Λογισμικού ΙΙ*, Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο
* Θραμπουλίδης Κλ. (2001), *Σχεδιασμός Λογισμικού Τόμος Γ΄ - Γλώσσες Προγραμματισμού ΙΙ (Αντικειμενοστρεφής Προγραμματισμός)*, Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο
* Ξένος Μ., Σταμέλος Ι., Φιτσιλής Π. (2008), *Σχεδιασμός Λογισμικού Τόμος Δ΄ - Προγραμματισμός Έργων Πληροφορικής – Αντικειμενοστρεφής Μεθοδολογίες*, Πάτρα: Ελληνικό Ανοικτό Πανεπιστήμιο
* Deitel, P., and Deitel, H. (2015). Java Προγραμματισμός. Εκδόσεις Μ. Γκιούρδας, 10η Έκδοση
* https://study.eap.gr, *Πρόσθετο υλικό Ενότητας ΠΛΗ24 και Γραπτές εργασίες προηγούμενων ετών*
* Δημιουργία και χρήση Derby DB (<https://www.youtube.com/watch?v=i-wYlPgtL2k&list=PL8Prx4IjOaZZyCJakQutExaMKg9g99Ton&index=5>)
* Scrum presentation (<https://www.youtube.com/watch?v=sb2_VewyHB4&list=PL8Prx4IjOaZZyCJakQutExaMKg9g99Ton&index=5>)
* Java Persistence API (JPA) Example

Πηγή: <https://www.youtube.com/watch?v=5jR0Ikpyqrg&t=42s>

* <https://stackoverflow.com/>
* <https://www.codejava.net/>
* <https://www.tutorialspoint.com/java/index.htm>
* <https://docs.oracle.com/javase/tutorial/>
* <https://www.w3schools.com/java/>
* <https://www.javatpoint.com/>
* <https://www.geeksforgeeks.org/java/>
* <https://github.com/LGoodDatePicker/LGoodDatePicker>
* <https://coderanch.com/>
* <https://www.journaldev.com/>
* Πανεπιστήμιο Θεσσαλίας – Τμήμα μηχανολόγων μηχανικών & Μηχανικών υπολογιστών

Πηγή: <https://courses.e-ce.uth.gr/ECE326/doku.php?id=java:toc>

**Βιβλία του εμπορίου:**

• Cay S. Horstmann - Big Java Late Objects (2012 edition)

• JavaNotesForProfessionals (StackOverflow’s Notes for professional edition)

Πηγή: <https://books.goalkicker.com/JavaBook/>

• Kathy Sierra, Bert Bates, Bert Bates - Head First Java, -O'Reilly Media (2005 edition)