ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

Εφαρμογή για εικονική εταιρία με όνομα The Eater's Club

Μέλη ομάδας:

1. Παπανάγνου Κωνσταντίνος (Υπεύθυνος) 4378
2. Μελισσός Αθανάσιος 4375
3. Ράπτη Έλλη 4319
4. Γκρούνοβα Ντενίτσα 4326

Επιτελική Αναφορά

Το πρόγραμμα μας ονομάστηκε The Eater's Club από το context το οποίο δημιουργήσαμε για την εταιρία που προορίζεται. Το πρόγραμμα μας αποτελείται από 4 οθόνες όπου 3 εκ των οποίων εμφανίζονται σαν ξεχωριστά παράθυρα. Η πρώτη διεπαφή που εμφανίζεται στον πελάτη είναι η διεπαφή η οποία περιέχει το λογότυπο της εταιρίας, την λίστα με το μενού, ένα combobox φίλτρου για την ευκολότερη αναζήτηση και επιλογή ανά κατηγορία (Το οποίο δουλεύει δυναμικά) και τις επιλογές προσθήκης στο καλάθι χαμηλά, καθώς και τα κουμπιά για μετάβαση στις άλλες διεπαφές. Το πρόγραμμα αντλεί δεδομένα από βάση δεδομένων Sqlite. Οι οθόνες στις οποίες μπορεί ο χρήστης να μεταβεί είναι η εμφάνιση του καλαθιού του και ολοκλήρωση πληρωμής. Στην διεπαφή του καλαθιού ο χρήστης βλέπει τα προϊόντα που έχει βάλει στο καλάθι και έπειτα μπορεί να επεξεργαστεί το καλάθι. Πατώντας το κουμπί ολοκλήρωση παραγγελίας μπορεί ο χρήστης να μεταβεί στην διεπαφή ολοκλήρωσης παραγγελίας όπου θα βάλει τα στοιχεία του για να ολοκληρώσει την παραγγελία. Υπάρχει επιλογή πληρωμής με κάρτα και μετρητά. Σε περίπτωση επιλογής μετρητών (Αντικαταβολή) η παραγγελία ολοκληρώνεται αμέσως. Στην αντίθετη περίπτωση επιλογής κάρτας ο χρήστης μεταβαίνει στην επόμενη σκηνή όπου μπορεί να βάλει τα στοιχεία της κάρτας με σκοπό να ολοκληρώσει την πληρωμή.

Σκοπός και Διαδικασία

Ο σκοπός της ομάδας μας είναι να μάθουμε όσο το δυνατόν περισσότερα μπορούμε σχετικά με την Java και την JavaFX. Η εφαρμογή αυτή προορίζεται σαν ένα front end για την δημιουργία παραγγελιών. Η διαφορά με το e-food είναι πως αυτή η εφαρμογή δεν τρέχει σε web browser αλλά τρέχει τοπικά στον υπολογιστή του χρήστη και θα επικοινωνεί με τον server του μαγαζιού για να κλείσει τις παραγγελίες. Το Project το χωρίσαμε σε 4 διεπαφές. Είμαστε 4 άτομα επομένως κάθε άτομο ανέλαβε και μία διεπαφή. Όσον αφορά το Backend και την βάση δεδομένων λόγω προγραμματιστικής εμπειρίας στο παρελθόν και για να διευκολύνω την ομάδα το ανέλαβα εγώ (Παπανάγνου Κωνσταντίνος). Στην πρώτη κλήση που κάναμε αποφασίσαμε το theme της εφαρμογής (Χρώματα και όλα τα αισθητικά), όπου ο καθένας είπε την ιδέα του και τελικά υλοποιήσαμε την καλύτερη ιδέα. Αυτό που επίσης συμφωνήσαμε είναι πως όλες οι διεπαφές θα ακολουθούν κατά σύμβαση το ολικό theme που επιλέξαμε στην αρχή. Κάθε μέλος δούλεψε ξεχωριστά, βοηθήσαμε όλοι όπου μπορούσαμε κάνοντας κλήσεις "brainstorming" για να σκεφτούμε πως μπορούμε να λύσουμε τα προβλήματα που αντιμετωπίζαμε. Αφού όλοι ολοκληρώσαμε τα κομμάτια μας τα ενώσαμε σε ένα project και τα κολλήσαμε όλα μαζί.

Σημείωση: Το τι έχει κάνει ο καθένας φαίνεται στην αναλυτική αναφορά όπου έχει το όνομα του ατόμου και ακολουθεί η αναφορά του με αυτά που ανέλαβε

Αναλυτικές Αναφορές κάθε μέλους:

ΠΑΠΑΝΑΓΝΟΥ ΚΩΝΣΤΑΝΤΙΝΟΣ 4378

Backend

Ανέλαβα να γράψω το backend της εφαρμογής το οποίο ήταν σχετικά απλό. Αποτελείται από το μοντέλο (κλάση) ενός «πιάτου» που προσφέρει το κατάστημα. Έχει πληροφορίες σχετικά με στοιχεία που προωθεί η εταιρία όπως το όνομα του πιάτου, την τιμή του πιάτου, την περιγραφή του πιάτου και εικόνα του πιάτου, αλλά και πληροφορίες metadata που χρησιμοποιεί η εφαρμογή για την πραγματοποίηση μιας παραγγελίας όπως την ποσότητα (Πόσα πιάτα θέλει ο πελάτης να πάρει). Ένα ερώτημα ήταν αν στην java μπορώ να περάσω by reference την λίστα για άμεση τροποποίηση. Η απάντηση που δόθηκε από μία αναζήτηση στο ιντερνετ είναι όχι και για αυτόν τον λόγο η λύση που σκέφτηκα ήταν να δημιουργήσω μία στατική λίστα καλαθιού μέσα στην δομή του «πιάτου»

Note: Τα σχόλια για την παραγγελία (π.χ. πόσο ψημένο θα είναι το κρέας, ή κάποια συγκεκριμένη παράκληση κατά την παράδοση) αποφασίστηκε να μην αποθηκεύονται σε δομή στο backend αλλά να στέλνονται κατευθείαν από το front end στο χαρτί παραγγελίας.

Στο backend της εφαρμογής συνδέεται επίσης μία βάση δεδομένων Sqlite μέσω της κλάσης DatabaseHandler που είναι πρακτικά μία wrapper κλάση για την διαχείριση της βάσης δεδομένων. Η βάση δεδομένων περιέχει τα πιάτα που προσφέρει το κατάστημα με όλα τα metadata τους και ενημερώνεται σε συχνή βάση από το κατάστημα (Θεωρητικά).

Στο backend συνάντησα ένα σημαντικό πρόβλημα με το Controller intercommunication. (Δεν μπορούσα να χρησιμοποιήσω static object από κλάση στο backend μεταξύ των Controllers για μεταφορά δεδομένων για κάποιο λόγο που δεν έμαθα ποτέ.) Ψάχνοντας στο internet βρήκα μια πιθανή λύση με Context αλλά δεν λειτουργούσε στο δικό μας project, πράγμα το οποίο με άφησε με μόνο μία λύση. Έπρεπε να κάνω bypass την javafx και αυτό το έκανα μέσω της Sqlite. Έφτιαξα ένα νέο πίνακα στην βάση δεδομένων για την προσωρινή αποθήκευση του καλαθιού και κάθε Controller θα τραβάει τα δεδομένα από την βάση. Αυτό το bypass παρότι θυσιάζει επεξεργαστική ισχύ, λύνει το πρόβλημα που αντιμετωπίζει η java.

Αρχική οθόνη

Επίσης ανέλαβα να γράψω την αρχική οθόνη η οποία είναι και η πρώτη που εμφανίζεται στον χρήστη που ξεκινάει την εφαρμογή μας. Η οθόνη έχει ένα πλήρως

custom flat design όπου τα χρώματα είναι επιλεγμένα σε σκούρο theme και τα γράμματα ανοιχτά. Η οθόνη περιέχει αρκετά γραφικά αντικείμενα, μερικά εκ των οποίων είναι custom που δημιουργήσαμε. Πιο συγκεκριμένα περιέχει:

- ListView Control
- Custom ListViewCell (ListViewItems)
- Button
- Labels
- Spinner
- Dialog
- Ένα custom κουμπί το οποίο αποτελείται από ένα HBox και σαν σώμα περιέχει ένα ImageView και ένα Label.
- ImageView (Logo)

Πέρα από τα γραφικά αντικείμενα η οθόνη αυτή περιέχει επίσης 4 animations.

- Ένα animation είναι στο logo όπου κάνει ένα rotate 180 μοιρών στον άξονα X κάθε φορά που ο χρήστης περνάει τον κέρσορα από πάνω του
- Ένα animation είναι στο brand name label το οποίο κάνει rotate αρνητικές 30 μοίρες στον άξονα Z κάθε φορά που ο χρήστης περνάει τον κέρσορα από πάνω του.
- Και τέλος τα τελευταία 2 animations βρίσκονται στα κουμπιά Το καλάθι μου και Ολοκλήρωση παραγγελίας τα οποία κάνουν την ίδια λειτουργία. Όποτε ο χρήστης περνάει τον κέρσορα από πάνω τους αυτά μεγαλώνουν λίγο και αλλάζουν το χρώμα τους προς μια πιο σκούρα απόχρωση.

Για να εμβαθύνω λίγο στον τρόπο λειτουργίας της εφαρμογής θα φέρω ένα παράδειγμα χρήσης. Μπαίνω σαν χρήστης στην εφαρμογή για να παραγγείλω. Ανοίγω την εφαρμογή, και βλέπω την αρχική οθόνη. Κατευθείαν έχω φορτωμένο στην λίστα μου όλο το μενού του μαγαζιού στην λίστα, με φωτογραφίες, περιγραφές και τιμές. Μπορώ να επιλέξω κάποιο πιάτο πατώντας επάνω του. Μόλις πατήσω πάνω του κάτω χαμηλά ενεργοποιείται το μενού επιλογής ποσότητας το οποίο μπορώ να τροποποιήσω για να προσθέσω περισσότερα από ένα πιάτα της ίδιας κατηγορίας. Μπορώ να τροποποιήσω την επιλογή μου μέσω του διαθέσιμου Spinner. Αυτόματα ενημερώνεται για κάθε επιπλέον πιάτο που προσθέτω στην παραγγελία μου το τελικό ποσό για αυτό το πιάτο. Μόλις πατήσω το κουμπί «Προσθήκη στο Καλάθι» το/τα πιάτο/α μου εισάγονται στο καλάθι μου και εμφανίζεται ένα μήνυμα επιτυχίας το οποίο περιέχει ενημερωτικά το καλάθι μου με τις τιμές, τις ποσότητες και την τελική τιμή της παραγγελίας μου. Αν θέλω να τροποποιήσω την παραγγελία μου μπορώ να πατήσω το κουμπί «Το καλάθι μου» το οποίο μου εμφανίζει μία οθόνη με το καλάθι μου στο οποίο μπορώ να αφαιρέσω ή να τροποποιήσω την παραγγελία μου. Μόλις κατασταλάξω σε αυτά που θέλω, γυρνάω στην αρχική οθόνη και πατάω ολοκλήρωση παραγγελίας. Εμφανίζεται ένα νέο παράθυρο το οποίο περιέχει την τελική μου

παραγγελία για επιβεβαίωση, τελική τιμή, ένα πλαίσιο κειμένου για τυχών σχόλια στην παραγγελία μου και τους τρόπους πληρωμής. Μπορώ να επιλέξω πληρωμή με μετρητά ή με χρεωστική/πιστωτική κάρτα. Στην περίπτωση μετρητών η παραγγελία ολοκληρώνεται άμεσα χωρίς περαιτέρω καθυστερήσεις. Στην περίπτωση κάρτας αλλάζει η οθόνη, σε μία οθόνη εισαγωγής των στοιχείων της κάρτας για ολοκλήρωση της παραγγελίας.

Custom ListViewCells

Τέλος ανέλαβα και έγραψα το ListViewCell το οποίο είναι πλήρως custom και δεν έχουμε δει στο μάθημα (Τουλάχιστον στις 16 Απριλίου που γράφω αυτό το κείμενο). Υπήρξαν πάρα πολλά προβλήματα κατά την δημιουργία τους και μια μεγάλη προγραμματιστική παρέμβαση. Απαιτήθηκε πολύ αναζήτηση στο ιντερνετ για να μπορέσει να ξεπεραστεί το πρόβλημα με το πώς λειτουργεί το ListViewCell και πώς να δουλέψει ο controller. Να σημειωθεί πως δεν υπήρξε καθόλου copy-paste κώδικα από το ιντερνετ σε αυτή τη φάση της εφαρμογής επειδή πολύ απλά δεν λειτουργούσε τίποτα από αυτά που υπήρχαν. Μετά από πολύ trial and error κατάφερα να το κάνω να λειτουργήσει αλλά με το layout να δημιουργείται μέσω κώδικα. Δεν κατάφερα ποτέ να ξεπεράσω το πρόβλημα με τα fxml αρχεία όπου για κάποιο λόγο δεν δεχόταν τον controller. Ασχέτως με τα τεχνικά προβλήματα αυτής της φάσης στο τέλος δούλεψε το γραφικό στοιχείο μου και χρησιμοποιήθηκε στην αρχική οθόνη την οποία έφτιαξα και στην οθόνη εμφάνισης του καλαθιού του χρήστη.

Για να κατανοήσουμε τον τρόπο λειτουργίας είναι αρκετό να αναδειχτούν αυτές οι συγκεκριμένες γραμμές κώδικα. Αυτό που κάνουμε εδώ είναι να δημιουργήσουμε μία κλάση MainListViewItemCell η οποία είναι το κάθε ένα ξεχωριστό item της λίστας μας. Φτιάχνουμε μέσω κώδικα κάθε κομμάτι του item ξεχωριστά όταν καλούμε τον constructor. Αυτή η κλάση όπως φαίνεται κληρονομεί από την κλάση ListCell με τύπο δεδομένων Plate (Αυτό ονομάζεται Generic Class και λειτουργεί για οτιδήποτε τύπο και να δώσουμε πράγμα που το κάνει πολύ ισχυρό και ευέλικτο για κάθε πιθανή χρήση) .

```
public class MainListViewItemCell extends ListCell<Plate>{
    private HBox root = new HBox();
    private ImageView PlateImageView = new ImageView();
    private Label PlateNameLbl = new Label();
    private Label PlatePriceLbl = new Label();
    private Label PlateDescriptionLbl = new Label();

    public MainListViewItemCell() {
        beautifyControls();
        configureLayout();
    }

    private void configureLayout() {
        root.getChildren().addAll(PlateImageView, createMasterContainer());
    }
```

Από την στιγμή που κληρονομούμε την κλάση ListCell μας δίνεται η δυνατότητα να κάνουμε Override την μέθοδο που περιέχει, και καλεί κάθε φορά που δημιουργείται το item μας, updateItem. Αυτή μας δίνει τις πληροφορίες για το τρέχων «πιάτο» από την λίστα και ένα Boolean που μας λέει αν υπάρχουν δεδομένα για να εμφανιστούν σε αυτό το κελί. Μπορεί το τρέχων κελί να χρησιμοποιείται σαν κενό (Αυτά τα λέει στο documentation της oracle). Αξίζει να σημειωθεί πως αυτή η μέθοδος καλείται αυτόματα από την δομή η οποία χειρίζεται τα items. Στην περίπτωση μας αυτή η δομή είναι η ListView. Επομένως με βάση όλα αυτά που είπαμε, αν έχουμε δεδομένα για να δείξουμε στο τρέχων κελί καλούμε την addContext μέθοδο για να τροποποιήσει τα ανάλογα γραφικά στοιχεία στο κελί μας.

```java
@Override
public void updateItem(Plate plate, boolean empty) {
    super.updateItem(plate, empty);

    if(empty) {
        clearContext();
    }
    else {
        addContext(plate);
    }

}


private void clearContext() {
    setText(null);
    setGraphic(null);
}


private void addContext(Plate plate)
{
    setText(null);
    File file = new File(plate.getImagePath());
    PlateImageView.setImage(new Image(file.toURI().toString()));
    PlateNameLbl.setText(plate.getPlateName());
    PlatePriceLbl.setText(String.format("%.2f",plate.getPrice()) + "€");
    PlateDescriptionLbl.setText(plate.Description());
    setGraphic(root);
}
```

Όμως το πραγματικό πρόβλημα δεν ήταν εδώ. Το δύσκολο είναι η επικοινωνία και δημιουργία των στοιχείων αυτών. Το setup της λίστας και των κελιών της φαίνεται στο παρακάτω screenshot:

```java
private void setupList(ObservableList<Plate> plates) {
    //Set and Populate ListView
    listView.setCellFactory((lv) -> { return new MainListViewItemCell();});
    listView.setItems(plates);
    listView.getSelectionModel().selectedItemProperty().addListener(new ChangeListener<Plate>() {
        public void changed(ObservableValue<? extends Plate> ov, final Plate oldValue, final Plate newValue) {
            selectedPlateLbl.setText("Plate: " + newValue.getPlateName());
            selectedPriceLbl.setText("Plate Price: " + String.format("%.2f", newValue.getPrice()) + "€");
            finalPriceLbl.setText("Final Plate Price: " + String.format("%.2f",newValue.getPrice() * quantitySpinner.getValue())+ "€");
            selected = newValue;
            quantitySpinner.setDisable(false);
        }
    });
}
```

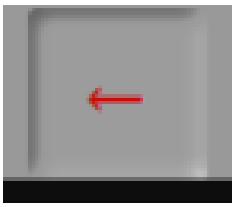
Αυτό το κομμάτι κώδικα παρότι μικρό ήταν αρκετά περίπλοκο την ώρα της αναζήτησης. Καταρχάς στην δομή ListView φτιάχνουμε ένα CellFactory το οποίο πρακτικά είναι ο τρόπος με τον οποίο θα φτιάχνει τα κελιά για κάθε στοιχείο της λίστας. (Ή τουλάχιστον αυτό κατάλαβα εγώ). Η αμέσως επόμενη γραμμή βάζει στην δομή την λίστα με τα πιάτα που έχει το κατάστημα. Αυτές οι δύο γραμμές κώδικα σε συνδυασμό μαζί κάνουν το όλο σύστημα να δουλεύει, καθώς για κάθε αντιείμενο της λίστας καλεί το CellFactory το οποίο περιγράφει πως θα δημιουργηθεί το κελί, και μετά η ListView εσωτερικά καλεί την μέθοδο updateItem(Plate plate, Boolean empty)

η οποία τροποποιεί τα γραφικά στοιχεία του κελιού για να γεμίσει με τις πληροφορίες του συγκεκριμένου «πιάτου»

Αθανάσιος Μελισσός 4375

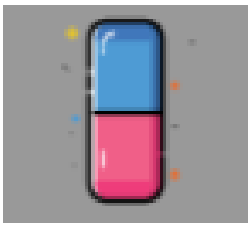
Περιγραφή οθόνης προεπισκόπησης παραγγελίας (καλάθι):

Ο σκοπός αυτής της οθόνης είναι να μπορεί ο χρήστης να κάνει μια σύντομη προεπισκόπηση της παραγγελίας του ,προτού περάσει στο τελευταίο βήμα της επιλογής τρόπου πληρωμής. Επομένως ο σκοπός αυτού του μέρους της εφαρμογής είναι να προτείνει κάτι συνοδευτικό και να του παριστά τα αντικείμενα που επέλεξε σε μια λίστα έτσι ώστε να έχει μια ξεκάθαρη εικόνα των αντικειμένων που επιλέχθηκαν, αποφεύγοντας έτσι τυχών λάθη πριν το τελικό βήμα. Πιο απλά αυτή η οθόνη λειτουργεί ως ¨καλάθι¨. Στο καλάθι έχουνε προστεθεί και εικονίδια με τα οποία ο χρήστης μπορεί να κατανοήσει καλύτερα την λειτουργικότητα και τον σκοπό κάποιου κουμπιού, για παράδειγμα: εάν ο χρήστης ξεχάσει να προσθέσει κάποιο ποτό για να συνοδέψει την παραγγελία του υπάρχει το κατάλληλο κουμπί με το οποίο μπορεί εύκολα και απλά να γυρίσει στο κατάλληλο μενού όπου μπορεί να προσθέσει το συνοδευτικό της αρέσκειας του. Το κουμπί που επιλέχθηκε θυμίζει το σύμβολο της επιστροφής και έχει τον κατάλληλο χρωματισμό έτσι ώστε να είναι ορατό από τον χρήστη, μάλιστα γύρω από το κουμπί προστέθηκε εφέ έτσι ώστε να διαφέρει από το υπόλοιπο backround. Ακολουθεί αναπαράσταση αυτού:

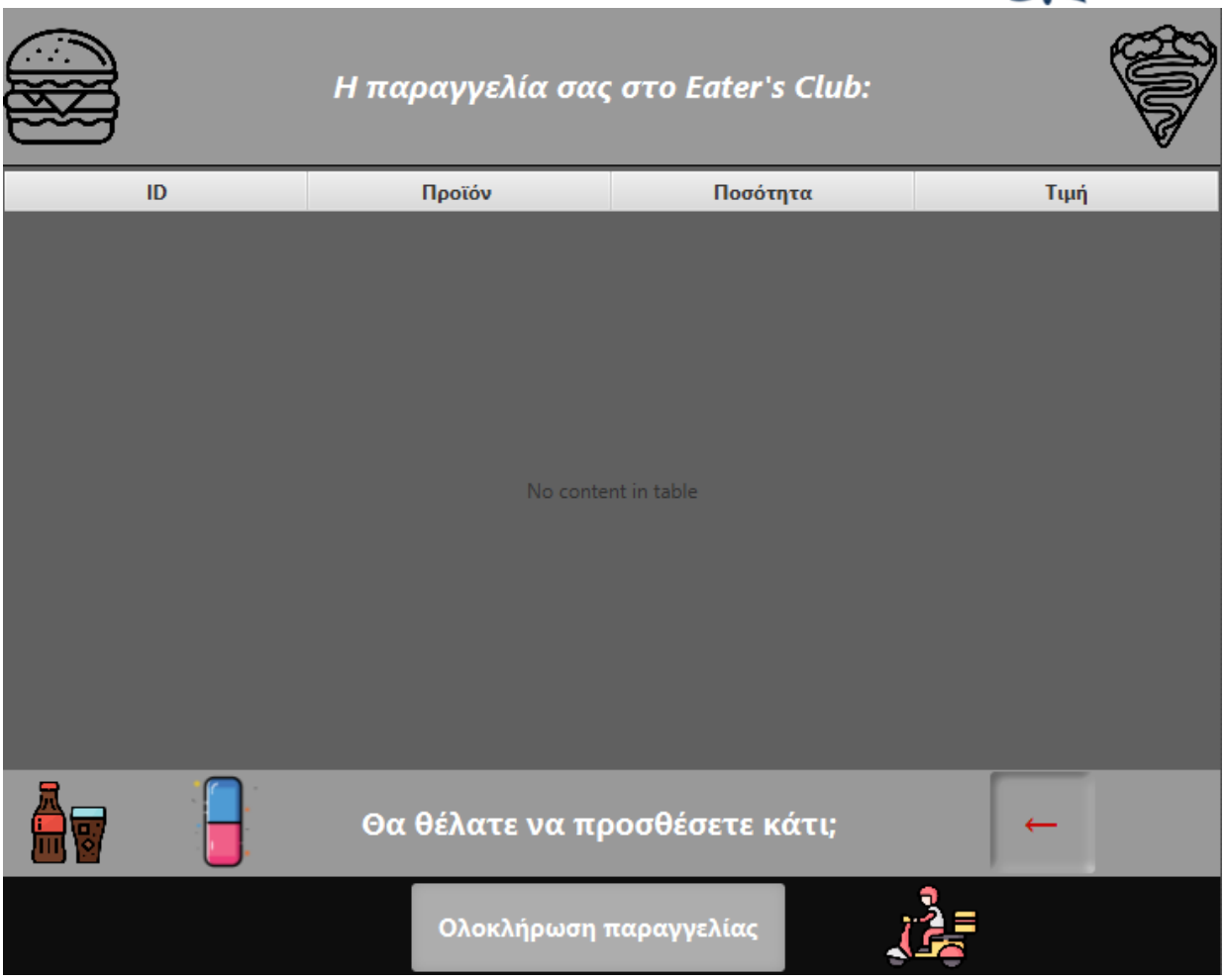


Επιπλέον, έγινε συνδυασμός με την αναπαράστασης μικρού εικονιδίου τύπου ανθρακούχου ποτού. Ακολουθεί αναπαράσταση του προαναφερόμενου εικονιδίου:

Να σημειωθεί πως, όπως και τα προηγούμενα εικονίδια, στο κουμπί αφαίρεσης παραγγελίας επιλέχθηκε κάτι αντιπροσωπευτικό επομένως επιλέχθηκε η εικόνα μιας γόμας.



Ακολουθεί αναπαράσταση της τελικής οθόνης καλαθιού όπου είναι ορατά τα εξής στοιχεία: Το κουμπί ολοκλήρωσης παραγγελίας το οποίο μεταφέρει τον χρήστη στη τελική οθόνη της πληρωμής, το κουμπί πρόσθεσης αντικειμένων τύπου αριστερού βέλους (πίσω), το κουμπί αφαίρεσης παραγγελίας με μορφή γόμας και τέλος η επικεφαλίδα της οθόνης ¨Η παραγγελία σας στο Eater’s Club¨:



Τέλος να σημειωθεί ότι κάθε κουμπί έχει συνοδευτεί με την κατάλληλη αναπαράσταση εικονιδίου, αυτό συμπεριλαμβάνει και το κουμπί ¨Ολοκλήρωση παραγγελίας¨ όπου είναι ορατό ένα εικονίδιο που αναπαριστά έναν διανομέα. Τα εικονίδια στο πάνω μέρος της οθόνης είναι καθαρά διακοσμητικά και δεν προσφέρουν κάποια λειτουργία.

Τα εικονίδια βρέθηκαν στην εξής ιστοσελίδα: www.flaticon.com .

Προγραμματιστικό μέρος οθόνης Cart

Μέρη οθόνης

Η λίστα αυτή εμπεριέχει τα χαρακτηριστικά από τα οποία αποτελείται η παραπάνω οθόνη όπως buttons, labels και άλλα τεχνικά χαρακτηριστικά τα οποία δεν αφορούνε αποκλειστικά την αμφίεση της.

Αυτά τα χαρακτηριστικά είναι:

- Button

- Label
- Line
- TableView με τρία TableColumn
- Vbox με 3 Hbox

Label:

Πιο αναλυτικά, τα label χρησιμοποιήθηκαν για την τήρηση απόστασης αντικειμένων στην οθόνη κρατώντας τα άδεια καθώς και για την αναγραφή του τίτλων στην οθόνη όπως το “The Eater’s Club”. Το επάνω μέρος της οθόνης όπου βρίσκεται ο τίτλος έχει χωριστεί με ένα απλό line μεταξύ του Hbox και TableView για να ξεχωρίζουν τα μέρη καλύτερα.

TableView:

Εκεί εμφανίζονται όλα τα χαρακτηριστικά της παραγγελίας του πελάτη η οποία αποτελείται από την ποσότητα το είδος και την τιμή του προϊόντος, με το πάτημα των label της λίστας γίνεται ανάλογη στοίχιση κατά αύξουσα ή κατά φθίνουσα σειρά. Το TableView τοποθετήθηκε στο AnchorPane για την καλύτερη διαχείριση του καθώς τα Vbox και Hbox δεν επέτρεπαν την εύκολη αλλαγή διαστάσεων. Τα TableColumn είναι επίσης στοιχισμένα με ίσα κενά μεταξύ τους ώστε να αποφευχθεί ο συνωστισμός τιμών λόγο μικρής χωρητικότητας.

Vbox:

Τα Vbox αποτελούνε το μεγαλύτερο μέρος της οθόνης καθώς εκεί βρίσκονται εικόνες, labels και κυρίως buttons τα οποία έχουνε τοποθετηθεί σε Hbox. Αρχικά το “πάνω” Hbox εμπεριέχει τις δύο διακοσμητικές εικόνες και τον τίτλο της οθόνης, διαχωρίζεται με ένα line μεταξύ με το παρακάτω TableView και τα χαρακτηριστικά έχουνε απόσταση χάρη σε 2 άδεια label. Το δεύτερο στη σειρά Hbox εμπεριέχει ξανά ένα label όπου αναγράφεται “Θα θέλατε να προσθέσετε κάτι;” μια εικόνα, δύο άδεια label για την καλύτερη απόσταση των χαρακτηριστικών που υπάρχουνε, επιπρόσθετα, το button που έχει τον ρόλο του back button έχει επεξεργαστεί έτσι ώστε να μην γίνεται “ένα” με το χρώμα του Hbox. Αυτό έγινε μέσω της αλλαγής χρώματος από το βελάκι και τα εφέ Lightning και Shadow δίνοντας έτσι μια καλύτερη αίσθηση βαθύτητας. Τέλος, το τρίτο Hbox εμπεριέχει 2 label για την απόσταση των χαρακτηριστικών μεταξύ τους και ένα Button όπου έχει επιλεχθεί το εφέ Glow για να ξεχωρίζει.

Παράδειγμα τύπου χρήσης οθόνης

Ο πελάτης αφότου έχει επιλέξει την αρεστή ποσότητα και είδος φαγητού επιθυμεί να προχωρήσει στο επόμενο βήμα της οριστικοποίησης ή την επισκόπηση αυτού. Για να το καταφέρει αυτό θα πρέπει να επιλέξει το κουμπί του cart έτσι ώστε να εμφανιστεί η ανάλογη οθόνη, σε αυτή θα μπορεί να δει την παραγγελία του μαζί με το είδος, τιμή και ποσότητα αυτού με την προαιρετική επιλογή της αντιστοίχησης. Έπειτα μπορεί, αφού σιγουρευτεί ότι είναι έτοιμος για την οριστικοποίηση της παραγγελίας του, να πατήσει "Ολοκλήρωση παραγγελίας" και να προχωρήσει στο τελικό βήμα της πληρωμής.

Προγραμματιστικό κομμάτι οθόνης

Στο προγραμματιστικό κομμάτι της οθόνης cart αρχικά προστέθηκαν όλα τα απαραίτητα λειτουργικά χαρακτηριστικά του controller. Σημαντικό είναι επίσης το ότι το generated όνομα υπέστη αλλαγή γι' αυτό δεν περιέχει την φράση "controller". Μιας και που επιλέχθηκε TableView αναγράφτηκαν πρώτα οι κατάλληλες εντολές ανάλογες με τα ονόματα της κολώνας που αντιπροσώπευαν, οι ίδιες διαδικασίες ακολούθησαν για τα κουμπιά εκτός αυτού της διαγραφής. Ακολουθεί απόσπασμα του προαναφερόμενου κώδικα:

```
public class Othoni_cart implements Initializable{
    @FXML Button BackButton;
    @FXML Button OrderCompletionBtn;
    @FXML
    private TableView<Order> FoodTables;
    @FXML
    private TableColumn<Order, String> ID;
    @FXML
    private TableColumn<Order, String> posotita;
    @FXML
    private TableColumn<Order, String> proion;
    @FXML
    private TableColumn<Order, String> timi;
```

Ακολουθεί η δημιουργία λειτουργιών των BackButton και OrderCompletionBtn όπου το ένα θα γυρνάει στην προηγούμενη οθόνη και το άλλο θα προχωράει στην επόμενη οθόνη πληρωμής "Payment.fxml"αντίστοιχα.

```
BackButton.setDisable(false);
BackButton.setOnAction(new EventHandler<ActionEvent>() {

    @Override public void handle(ActionEvent event) {
    ((Stage)BackButton.getScene().getWindow()).close();
    }
});
```

(BackButton το οποίο κλείνει το παράθυρο)

```
OrderCompletionBtn.setDisable(false);
OrderCompletionBtn.setOnAction(new EventHandler<ActionEvent>() {

    @Override public void handle(ActionEvent event) {
        try {
            Stage window = new Stage();
            Parent root = FXMLLoader.load(getClass().getResource("Payment.fxml"));
            window.getIcons().add(new Image("file:./src/img/logo black.png"));
            window.setTitle("The Eater's Club");
            Scene scene = new Scene(root);

            scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());
            window.setResizable(false);
            window.setScene(scene);
            Main.OpenedStage = window;
            window.showAndWait();
        } catch(Exception el) {
            el.printStackTrace();
            System.out.println(el.getMessage());
        }
    }
});
```

(OrderCompletionBtn το οποίο προχωράει στην οθόνη πληρωμής)

Ακολουθεί μέθοδος διαγραφής παραγγελίας που εκτελείται με το κουμπί "RemoveItem" όπου ο χρήστης επιλέγει την παραγγελία που θέλει να διαγράψει, έπειτα πατάει το κουμπί και πραγματοποιείται η διαγραφή της. Να σημειωθεί ότι ο χρήστης μπορεί να διαγράψει πολλαπλές παραγγελίες εάν επιθυμεί.

```
@FXML
private void removeClick(MouseEvent e) {
    for(Order order : FoodTables.getSelectionModel().getSelectedItems()) {
        FoodTables.getItems().remove(order);
        db.deleteFromCart(Integer.parseInt(order.getID()));
    }
}

public void initialize(URL location, ResourceBundle resources) {
    FoodTables.getSelectionModel().setSelectionMode(
            SelectionMode.MULTIPLE
        );
```

```
    ID.setCellValueFactory(new PropertyValueFactory<Order, String>("ID"));
    posotita.setCellValueFactory(new PropertyValueFactory<Order, String>("posotita"));
    proion.setCellValueFactory(new PropertyValueFactory<Order, String>("proion"));
    timi.setCellValueFactory(new PropertyValueFactory<Order, String>("timi"));

    ObservableList<Order> oblist = FXCollections.observableArrayList();

    for(Order order : db.getCart()) {
        oblist.add(order);
    }



    FoodTables.setItems(oblist);

}
```

Τέλος, ορατό είναι το βήμα 2 του πρώτου ScreenShot όπου μετά την δημιουργία των TableColumn σε αυτό το ScreenShot όπου ολοκληρώνεται η δημιουργία και ένωση των Values με το ανάλογο όνομα που τους δόθηκε. Ορατός είναι επίσης ο τρόπος με τον οποίο το TableView θα παίρνει τις τιμές του και θα προβάλλονται στον χρήστη.

Ντενίτσα Γκρούνοβα 4326

3η Σκηνή:Ολοκλήρωση παραγγελίας

Το κομμάτι που ανέλαβα είναι η ολοκλήρωση της παραγγελίας. Ο πελάτης αφού έχει επιλέξει τα προϊόντα που θέλει να παραγγείλει ανάλογα με το που βρίσκεται είτε στην αρχική οθόνη, είτε στο καλάθι με τα προϊόντα που έχει επιλέξει, υπάρχει ένα Button που τον οδηγεί στην ολοκλήρωση της παραγγελίας. Αφού το πατήσει εμφανίζεται η σκηνή που αφορά την Ολοκλήρωση Παραγγελίας.

Αρχικά, ξεκίνησα να δημιουργώ την διεπαφή από το πρόγραμμα SceneBuilder.Έστησα την οθόνη έτσι όπως ήθελα να εμφανίζεται και στη συνέχεια μέσω του eclipse και της javaFX έκανα την σκηνή μου λειτουργική.

Στην σκηνή όπου ο πελάτης ολοκληρώνει την παραγγελία του, βλέπουμε στα αριστερά ένα Table View που δείχνει έναν κωδικό ID,τα προϊόντα που έχει επιλέξει να παραγγείλει, την ποσότητα από αυτά και την αντίστοιχη τιμή τους. Από κάτω ακριβώς υπάρχει ένα Label που δείχνει το Σύνολο πληρωμής. Στα δεξιά υπάρχουν τα πεδία που πρέπει να συμπληρώσει ο πελάτης με τα προσωπικά του στοιχεία, το Ονοματεπώνυμο του, Τηλέφωνο, Email, Διεύθυνση, Όροφο και Κουδούνι. Ακόμη, έχω βάλει ένα TextArea για να προσθέσει εκεί κάποια οδηγία ή παρατήρηση για κάποιο από τα προϊόντα που επιθυμεί ή οτιδήποτε άλλο θεωρεί χρήσιμο για την καλύτερη εξυπηρέτηση του. Αν έχει συμπληρώσει τα πεδία, στη συνέχεια πρέπει να επιλέξει τον τρόπο πληρωμής. Υπάρχει το κατάλληλο Label<<Τρόπος πληρωμής>>

στα αριστερά και από κάτω δύο RadioButtons <<Μετρητά>> και <<Πιστωτική Κάρτα>> με τα <<Μετρητά>> να είναι επιλεγμένα. Από κάτω ακριβώς έχω βάλει ένα Button <<ΠΙΣΩ>> που του επιτρέπει να επιστρέψει πίσω και να προσθέσει αν θέλει κάτι και στα δεξιά άλλο ένα Button που γράφει <<ΠΛΗΡΩΜΗ>> και ολοκληρώνει την παραγγελία του. Αν έχει επιλέξει να πληρώσει με μετρητά, έχει συμπληρώσει υποχρεωτικά όλα τα πεδία(αν δεν το έχει κάνει την στιγμή που πατάει <<Πληρωμή>> εμφανίζεται το κατάλληλο μήνυμα στην οθόνη που του λέει να συμπληρώσει το πεδίο που έχει παραλείψει) τότε έχει ολοκληρώσει την παραγγελία του και πάνω από το TableView φορτώνει και εμφανίζεται ένα ProgressIndicator και του δείχνει ότι η παραγγελία έχει ολοκληρωθεί και πλέον το Button <<ΠΛΗΡΩΜΗ>> γράφει <<Έξοδος>> και αφού το πατήσει κλείνει το παράθυρο.

Υπάρχει και άλλη μία περίπτωση. Τι γίνεται αν ο πελάτης επιλέξει να πληρώσει με πιστωτική κάρτα. Πάλι πρέπει να έχει συμπληρώσει όλα τα πεδία με τα στοιχεία του και στη συνέχεια στον Τρόπο πληρωμής να επιλέξει το RadioButton που γράφει <<Πιστωτική Κάρτα>>. Αφού το επιλέξει πατάει το Button κάτω δεξιά ΠΛΗΡΩΜΗ και του ανοίγει την επόμενη σκηνή που αφορά την πληρωμή με Πιστωτική Κάρτα.

Τα components που χρησιμοποίησα στον SceneBuilder είναι τα εξής:

1. Από τα Containers:
    - AnchorPane
    - VBox
    - HBox

2. Από τα Controls:
    - Button
    - Label
    - ProgressIndicator
    - RadioButton
    - TableView
    - TableColumn
    - TextArea
    - TextField

Πιο συγκεκριμένα:

1. Επέλεξα το φόντο μου να είναι σκουρόχρωμο με άσπρα γράμματα σύμφωνα με τα χρώματα των προηγούμενων σκηνών επιλέγοντας χρώμα για το φόντο το #606060.
2. Έβαλα τα κατάλληλα στοιχεία ,έδωσα τις απαραίτητες αποστάσεις μεταξύ των TextFields από το μενού στα δεξιά→Properties→Margin

3. Έβαλα τα κατάλληλα Label και έδωσα τις ονομασίες σε κάθε Button,Label, TableView, TableColumn κλπ. και τα κατάλληλα id για να μπορέσω να τα χρησιμοποιήσω στην συνέχεια στον κώδικα.

Έπειτα, αφού κατέβασα το project E-FastFood που ξεκίνησαν οι συνάδελφοι μου με τις πρώτες σκηνές, πριν από την δικιά μου, από το GitHub (το χρησιμοποιήσαμε σαν εργαλείο κατά την διάρκεια της εργασίας και σε κάθε αλλαγή που κάναμε στον κώδικα μας ανανεώναμε το πρότζεκτ) έφτιαξα το αρχείο Payment.fxml είχα συνδέσει από το SceneBuilder τον controller και αποθήκευα κάθε φορά τις αλλαγές που έκανα. Μετά δημιούργησα το PaymentController.java δηλαδή την main του.

Στην PaymentController.java δημιουργώ το ToggleGroup και βάζω σε λειτουργία τα RadioButtons, τα Buttons ,φορτώνω στο TableView τα στοιχεία δημιουργώντας την Order.java όπως δείξαμε στο μάθημα. Εδώ συγκεκριμένα αντιμετώπισα πρόβλημα γιατί αρκετές φορές έτρεχα το πρόγραμμα και δεν φορτώνανε γιατί είχα δώσει άλλη ονομασία στον SceneBuilder. Με λίγη παρατηρητικότητα το βρήκα και το άλλαξα. Επίσης έφτιαξα την progress,έβαλα περιορισμούς στα TextFields, δηλαδή να μην μπορεί να κάνει πληρωμή εάν δεν έχει συμπληρώσει όλα τα πεδία και εδώ εμφανίζεται ένα παράθυρο που τον ενημερώνει για αυτό. Ακόμη έβαλα τις κατάλληλες εντολές για να πηγαίνει ο πελάτης στο κομμάτι που η πληρωμή θα γίνεται με πιστωτική κάρτα και όχι με μετρητά. Την μέθοδο showDialog με το buttonOK την πήρα από τον συνάδελφο μου τον Κωνσταντίνο Παπανάγνου και την βάλαμε και σε άλλες σκηνές.

Έλλη Ράπτη 4319

Εφαρμογή Παραγγελιοληψίας

- Ο χρήστης θα μπορεί να ολοκληρώνει την παραγγελία με κάποιον τρόπο πληρωμής. Στην περίπτωση κάρτας η διαδικασία θα πρέπει να φαίνεται.

Ο οθόνη που σχεδίασα είναι η τελική οθόνη της εφαρμογής. Εφόσον ο χρήστης έχει επιλέξει ότι θα πληρώσει με κάρτα σχεδίασα την οθόνη που θα εμφανίζεται στην οθόνη του χρήστη και θα ζητάει στοιχεία σχετικά με την κάρτα, δηλαδή αν είναι visa, paypal ή mastercard. Επίσης θα χρειάζονται:

1. Ο αριθμός της κάρτας.
2. Η ημερομηνία λήξης
3. Ο τριψήφιος αριθμός ασφαλείας που βρίσκεται στο πίσω μέρος κάθε κάρτας.
4. Το όνομα κατόχου

Αρχικά, έψαξα να βρω σχετικές εφαρμογές για να αποφασίσω και να πάρω ιδέες για το πώς θα στηθεί η οθόνη στον SceneBuilder. Έπειτα από πλοήγηση σε 2 εφαρμογές αποφάσισα να ακολουθήσω έναν σχετικά απλό σχεδιασμό.

Χρησιμοποίησα τα εξής στοιχεία από τον SceneBuilder ώστε να στήσω την οθόνη:

1. 1 περιέκτη 500x500
2. 3 Radio Buttons
3. 3 Text Fields
4. 1 Date Picker
5. 1 Button
6. 2 Label
7. Progress Indicator/ Progress Bar

Εφόσον έστησα την τελική μορφή της οθόνης μου στον SceneBuilder άλλαξα τα χρώματα στο φόντο και στις λέξεις έτσι ώστε να ταιριάζει με της υπόλοιπης ομάδας όπως τα είχαμε συμφωνήσει.

Έπειτα στο κομμάτι του κώδικα δεν έχει κάποιο κομμάτι που να με δυσκόλεψε ιδιαίτερα ότι χρειάστηκα ανέτρεξα στους κώδικες του μαθήματος και σε πηγές στο διαδίκτυο.

Ξεκίνησα κάνοντας inject τα buttons που μου χρειάστηκαν από τον SceneBuilder και ενώνοντας τα με τις αντίστοιχες ονομασίες μέσα στον κώδικα.

Έπειτα όρισα τα μηνύματα που θα εμφανίζονται στην οθόνη του χρήστη σε περίπτωση που δεν εισαχθούν σωστά το ονοματεπώνυμο, ο αριθμός κάρτας και ο τριψήφιος κωδικός. Όρισα επίσης το πεδίο του ονόματος να μην μπορεί να μείνει κενό, δηλαδή σε περίπτωση που δεν υπάρχει κανένας χαρακτήρας θα εμφανίζεται «Λανθασμένο Ονοματεπώνυμο». Αντίστοιχα, στο πεδίο του τριψήφιου κωδικού πρέπει να εισαχθούν αυστηρά 3 χαρακτήρες και στον αριθμό κάρτας αυστηρά 16. Σε οποιαδήποτε άλλη περίπτωση θα εμφανίζονται τα μηνύματα «Λανθασμένος αριθμός κάρτας» και «Λανθασμένος τριψήφιος κωδικός». Σε περίπτωση που όλα τα στοιχεία έχουν εισαχθεί σωστά θα εμφανίζεται το μήνυμα «Επιτυχής Πληρωμή».

Στη συνέχεια, όρισα στο Label του συνόλου να εμφανίζεται ένας τυχαίος αριθμός ως το σύνολο που πρέπει να πληρωθεί.

Ακόμα, δεν ήθελα ο DatePicker να εμφανίζει κενό πλαίσιο οπότε όρισα να δείχνει την σημερινή ημερομηνία κάθε φορά.

Αναφορικά με τα RadioButton χρησιμοποίησα ένα ToggleGroup έτσι ώστε ο χρήστης να επιλέγει μια κάρτα κάθε φορά αφού χωρίς αυτήν την κατηγοριοποίηση σε ToggleGroup ο χρήστης μπορούσε να επιλέξει περισσότερες από μία κάρτες. Με αυτόν τον τρόπο όμως στο τέλος κάθε παραγγελίας έχει τη δυνατότητα να επιλέξει μία μόνο κάρτα.

Ένα επιπλέον θέμα που προέκυψε όταν ολοκλήρωσα τα παραπάνω ήταν ότι μετά από κάθε παραγγελία έπρεπε να αδειάζει το καλάθι έτσι ώστε να είναι έτοιμη η εφαρμογή να χρησιμοποιηθεί και πάλι από τον χρήστη. Έτσι μετά από κάθε παραγγελία το καλάθι αδειάζει.

Τέλος, το κομμάτι που με δυσκόλεψε περισσότερο ήταν το ProgressBar γιατί δεν λειτουργούσε όταν έτρεχα το πρόγραμμα. Αφού το έψαξα στο internet βρήκα αντίστοιχο κομμάτι κώδικα. Όρισα ότι θέλω να εμφανίζεται στην οθόνη για τρία δευτερόλεπτα μέχρι δηλαδή η διαδικασία της πληρωμής να ολοκληρωθεί στο 100%. Αυτό που με δυσκόλεψε ήταν να καταλάβω ότι η τιμή που έβαζα έπρεπε να είναι αρνητική ώστε να τρέχει το animation. Εφόσον η διαδικασία ολοκληρωθεί εμφανίζεται το μήνυμα «Done» στην οθόνη.

Προτάσεις και Συμπεράσματα.

Η εργασία αυτή ήταν οργανωμένη από την αρχή της μέχρι το τέλος με κάθε μέλος να συνεισφέρει το κομμάτι του χωρίς ιδιαίτερες καθυστερήσεις. Από αυτή την εργασία κάθε μέλος κράτησε τα δικά του συμπεράσματα σχετικά με την τεχνολογία που χρησιμοποιήθηκε και τον τρόπο με τον οποίο δουλέψαμε. Υπάρχουν μέλη που τους άρεσε η τεχνολογία που χρησιμοποιήθηκε και υπάρχουν και άλλοι που θα προτιμούσαν κάποια άλλη τεχνολογία για μελλοντικό project. Ως επί το πλείστον η εργασία ολοκληρώθηκε με επιτυχία με όλα τα μέλη ευχαριστημένα με το συνολικό αποτέλεσμα της εργασίας.

Πηγές αναζήτησης και βοήθειας στο ιντερνετ:

<u>Αρχική οθόνη και Backend</u>

JavaFX HBox Alignment. (2015, April 17). Stack Overflow. https://stackoverflow.com/questions/29707882/javafx-hbox-alignment

Is Java “pass-by-reference” or “pass-by-value”? (2008, September 2). Stack Overflow. https://stackoverflow.com/questions/40480/is-java-pass-by-reference-or-pass-by-value

Customize ListView in JavaFX with FXML. (2013, October 25). Stack Overflow. https://stackoverflow.com/questions/19588029/customize-listview-in-javafx-with-fxml

KeepToo. (2020, September 4). How to create a custom list in JavaFX. YouTube. https://www.youtube.com/watch?v=4rVr_VT-4Z4

How can I show an image using the ImageView component in javafx and fxml? (2014, March 28). Stack Overflow. https://stackoverflow.com/questions/22710053/how-can-i-show-an-image-using-the-imageview-component-in-javafx-and-fxml

How to create a Dialog in JavaFX? (n.d.). Tutorials Point. https://www.tutorialspoint.com/how-to-create-a-dialog-in-javafx

Dialog (JavaFX 8). (2015, February 10). Oracle Documentation. https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/Dialog.html

How do you attach a listener to a JavaFX spinner? (2015, April 28). Stack Overflow. https://stackoverflow.com/questions/29926428/how-do-you-attach-a-listener-to-a-javafx-spinner

Define a relative path of image in Java FX. (2014, December 12). Stack Overflow. https://stackoverflow.com/questions/27446360/define-a-relative-path-of-image-in-java-fx

javafx Rotate Transition - javatpoint. (n.d.). Www.Javatpoint.Com. https://www.javatpoint.com/javafx-rotate-transition

Από εδώ τροποποιήθηκε το εξής μέρος κώδικα και χρησιμοποιήθηκε:

```
//Instantiating RotateTransition class
RotateTransition rotate = new RotateTransition();

//Setting Axis of rotation
rotate.setAxis(Rotate.Z_AXIS);

// setting the angle of rotation
rotate.setByAngle(360);

//setting cycle count of the rotation
rotate.setCycleCount(500);

//Setting duration of the transition
rotate.setDuration(Duration.millis(1000));

//the transition will be auto reversed by setting this to true
rotate.setAutoReverse(true);

//setting Rectangle as the node onto which the
ansition will be applied
rotate.setNode(rect);

//playing the transition
rotate.play();
```

Πηγές αναζήτησης και βιβλιογραφία που χρησιμοποιήθηκε για την οθόνη cart:

Table setup για TableView και TableColumn που το αποτελούν:

Code Amir

https://www.youtube.com/watch?v=A5fQbsJ-iF8&ab_channel=CodeAmirCodeAmir

Υποβοήθηση για database setup:

Rashid Iqbal

https://www.youtube.com/watch?v=LoiQVoNil9Q&ab_channel=HumayunKabirHumayunKabir

Υπόλοιπη υλοποίηση και SceneBuilder έγινε με την βοήθεια των μαθημάτων και τις σημειώσεις του καθηγητή.

Τρίτη Σκηνή Ολοκλήρωση Παραγγελίας

- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxButtonExample.java
- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxLabelExample.java
- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxRadioButtonExample.java
- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxTextAreaExample.java
- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxTextFieldExample.java
- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/CONTROLS/FxToggleButtonExample.java

- https://users.auth.gr/dmargoun/FX/SOURCE%20CODE/

  Ex28A_JDBC_PART_A.zip
  Ex28B_FinalPoject.zip

- https://stackoverflow.com/questions/9966136/javafx-periodic-background-task?fbclid=IwAR13AWNMXsktLzCWdTGZ6LjqZ5gJgaZvj5a-0T0UxfnBOwEQbShOYyFGXMM

- https://www.youtube.com/watch?v=A5fQbsJ-iF8
- **private void** showDialog από τον συνάδελφο Κωνσταντίνος Παπανάγνου.

Τέταρτη Σκηνή Ολοκλήρωσης Παραγγελίας με Κάρτα

- Σχετικά με τον Date Picker:
  https://stackoverflow.com/questions/36968122/how-to-set-javafx-datepicker-value-correctly?fbclid=IwAR2VjrgEAvsWQbqn-vUjUbHREeFHKQp_V09oHoQt5aIC6H_QY-eFljqGCho

- Σχετικά με το Progress Bar:
  https://stackoverflow.com/questions/9966136/javafx-periodic-background-task?fbclid=IwAR3pfVRzTyWOUNqx7MpKsJQRiyNTDMMj_S9d5aDqzrtAzG3nHZn3sEL-xys

- Κώδικας Μαθήματος

- Στο κομμάτι του κώδικα για την εμφάνιση των παραθύρων μας επέτρεψε ο Κωνσταντίνος Παπανάγνου , μέλος της ομάδας, να χρησιμοποιήσουμε δικό του κώδικα.