**Πανεπιστήμιο Πελοποννήσου**

**Σχολή Οικονομίας και Τεχνολογίας**

**Τμήμα Πληροφορικής και Τηλεπικοινωνιών**

**Π.Μ.Σ. στην Επιστήμη Υπολογιστών**

**Μεταπτυχιακή Εργασία**

**<Τίτλος>**

**Βενέρης Νικόλαος Δημήτριος**

**ΑΜ: 2022202002003**

**Πέππας Κωνσταντίνος**

**<Τρίπολη, Σεπτέμβρης 2023>**

**Πίνακας Περιεχομένων / Table of Contents**

[**II.** **Ευρετήριο Εικόνων / Table of Figures** 3](#_Toc145807169)

[**III.** **Εκτεταμένη Περίληψη** 3](#_Toc145807170)

[**IV.** **Extended abstract** 3](#_Toc145807171)

[**1.** **Εισαγωγή / Introduction** 3](#_Toc145807172)

[Η Ψηφιακή Επανάσταση 3](#_Toc145807173)

[Η Εξάπλωση των Εφαρμογών 3](#_Toc145807174)

[Στόχος Εργασίας 3](#_Toc145807175)

[**2.** **Δομικά στοιχεία της εφαρμογής** 4](#_Toc145807176)

[Η γλώσσα προγραμματισμού Dart 4](#_Toc145807177)

[**3.** **Περιγραφή της Εφαρμογής** 4](#_Toc145807178)

[Είσοδος στην εφαρμογή 5](#_Toc145807179)

[Προβολή λίστας προπονητικών προγραμμάτων/προπονήσεων 7](#_Toc145807180)

[Δημιουργία/ επεξεργασία προπονητικού προγράμματος 10](#_Toc145807181)

[Πραγματοποίηση προπόνησης. 13](#_Toc145807182)

[Λίστα ασκήσεων 16](#_Toc145807183)

[Διαχείριση του Ιστορικού 17](#_Toc145807184)

[Επεξεργασία Προσωπικού Προφίλ 18](#_Toc145807185)

[**V.** **Συμπεράσματα – Μελλοντικές κατευθύνσεις / Conclusions – Future Directions** 20](#_Toc145807186)

[**VI.** **Βιβλιογραφία / References** 20](#_Toc145807187)

[**VII.** **Παραρτήματα / Appendixes** 20](#_Toc145807188)

# **Ευρετήριο Εικόνων / Table of Figures**

[Εικόνα 1 - Σελίδα σύνδεσης. 6](#_Toc145807238)

[Εικόνα 2 - Αρχικοποίηση δεδομένων. 7](#_Toc145807239)

[Εικόνα 3 - Εγγραφή βάσης δεδομένων. 7](#_Toc145807240)

[Εικόνα 4 - Κενό ιστορικό προπονήσεων. 8](#_Toc145807241)

[Εικόνα 5 - Λίστα προπονήσεων. 9](#_Toc145807242)

[Εικόνα 6 - Επιπλέον επιλογές προπόνησης. 10](#_Toc145807243)

[Εικόνα 7 - Μήνυμα διαγραφής προπόνησης. 11](#_Toc145807244)

[Εικόνα 8 - Δημιουργία νέας προπόνησης. 12](#_Toc145807245)

[Εικόνα 9 - Προπόνηση με συμπληρωμένα πεδία. 13](#_Toc145807246)

[Εικόνα 10 - Μήνυμα σφάλματος. 14](#_Toc145807247)

[Εικόνα 11 - Έναρξη προπόνησης. 15](#_Toc145807248)

[Εικόνα 12 - Σελίδα πραγματοποίησης προπόνησης. 15](#_Toc145807249)

[Εικόνα 13 - Δήλωση άσκησης ως ολοκληρωμένη. 16](#_Toc145807250)

[Εικόνα 14 - Χρονόμετρο αποθεραπείας. 16](#_Toc145807251)

[Εικόνα 15 - Λίστα ασκήσεων. 17](#_Toc145807252)

[Εικόνα 16 - Ιστορικό προπονήσεων με εγγραφές. 18](#_Toc145807253)

[Εικόνα 17 - Επιπλέον πληροφορίες από το ιστορικό προπονήσεων. 19](#_Toc145807254)

[Εικόνα 18 - Σελίδα χρήστη. 20](#_Toc145807255)

[Εικόνα 19 - Σελίδα επεξεργασίας προσωποποιημένων δεδομένων χρήστη. 21](#_Toc145807256)

# **Εκτεταμένη Περίληψη**

# **Extended abstract**

# **Εισαγωγή / Introduction**

Στην εποχή μιας τεράστιας τεχνολογικής προόδου, οι κινητές συσκευές έχουν γίνει οι μηχανισμοί μέσω των οποίων έχουμε πρόσβαση, αλληλεπιδρούμε και εξατομικεύουμε τον ψηφιακό μας κόσμο. Αυτή η βαθιά μεταμόρφωση έχει αναδείξει την ανάπτυξη κινητών εφαρμογών στο προσκήνιο του ψηφιακού τοπίου, καθορίζοντας τον τρόπο που ζούμε, εργαζόμαστε, και συνδεόμαστε. Καθώς βρισκόμαστε στην αρχή της εποχής των κινητών, αυτή η εισαγωγή εξερευνά τον πολύπλοκο κόσμο της ανάπτυξης κινητών εφαρμογών, αναδεικνύοντας τη σημασία της, την εξέλιξή της και τις απεριόριστες δυνατότητες που προσφέρει.

## Η Ψηφιακή Επανάσταση

Η έναρξη της ανάπτυξης κινητών συσκευών σηματοδότησε μια επανάσταση που διαμόρφωσε οριστικά τον τρόπο που αλληλεπιδρούμε με την τεχνολογία. Οι κινητές συσκευές, από τα πρώτα κινητά τηλέφωνα έως τα εξελιγμένα smartphones και tablets της σήμερον ημέρας, έχουν εξελιχθεί από απλά εργαλεία επικοινωνίας σε πολυδιάστατους πύργους παραγωγικότητας, ψυχαγωγίας, και πληροφόρησης. Έχουν γεφυρώσει γεωγραφικές αποστάσεις, ξεπεράσει χρονικούς περιορισμούς, και έχουν ανοίξει νέες διαστάσεις της ανθρώπινης αλληλεπίδρασης.

## Η Εξάπλωση των Εφαρμογών

Στην καρδιά αυτής της επανάστασης βρίσκεται η εξάπλωση των κινητών εφαρμογών, μικρά κομμάτια λογισμικού που εξυπηρετούν μια πληθώρα αναγκών και επιθυμιών. Οι εφαρμογές έχουν εισάγει ένα επίπεδο εξατομίκευσης και διευκόλυνσης που κάποτε φαινόταν αδιανόητο. Έχουν μας επιτρέψει να κουβαλήσουμε τη δύναμη του υπολογισμού στην τσέπη μας και έχουν αναδιαμορφώσει βιομηχανίες, από την υγεία μέχρι την εκπαίδευση και τη διασκέδαση και έχουν κάνει τομείς όπως αυτός της υγείας πολύ πιο προσβάσιμους. Έτσι, ο κόσμος μπορεί να αναπτύσσει τις γνώσεις του συνεχώς και με μεγάλη ευκολία, να έχει περισσότερες επιλογές διασκέδασης, αλλά το σημαντικότερο, έχει την δυνατότητα να ενημερώνεται και να κάνει ενέργειές για την σωστή διατήρηση της υγείας του!

## Στόχος Εργασίας

Ο στόχος αυτής της εργασίας επικεντρώνεται στη δημιουργία μιας χρήσιμης για Android και Ios που αφορά την ανάπτυξη ενός "Workout Tracker". Η εφαρμογή αυτή έχει ως στόχο να προσφέρει στους χρήστες τη δυνατότητα να καταγράφουν, να παρακολουθούν και να βελτιώνουν την φυσική τους κατάσταση μέσω προπονητικών προγραμμάτων με τα οποία θα μπορούν να αλληλεπιδρούν.

Πιο συγκεκριμένα, ο χρήστης θα μπορεί να συνδέεται στην εφαρμογή, δίνοντάς του την δυνατότητα να δημιουργεί προσαρμοσμένα προπονητικά προγράμματα, ανάλογα με τις δικές του προτιμήσεις και ανάγκες. Αυτό σημαίνει ότι ο χρήστης μπορεί να δημιουργεί πλήρη προπονητικά προγράμματα, και να προσαρμόζει το περιεχόμενο ανάλογα με τους στόχους του.

Επιπλέον, η εφαρμογή θα προσφέρει στον χρήστη την δυνατότητα να διατηρεί ένα ιστορικό των προπονητικών προγραμμάτων που έχει ακολουθήσει στο παρελθόν, παρέχοντας τη δυνατότητα να ανατρέχει στην πρόοδό του. Αυτό θα επιτρέπει στον χρήστη να παρακολουθεί την εξέλιξη της φυσικής του κατάστασης, καταγράφοντας βάρη, επαναλήψεις και χρόνο που αφιερώνει σε κάθε άσκηση.

Τέλος, ο χρήστης θα έχει τη δυνατότητα να προσαρμόζει το προφίλ του στις ατομικές του ανάγκες και προτιμήσεις, προσθέτοντας πληροφορίες σχετικά με αυτόν.

Πιο συγκεκριμένα, οι στόχοι της εργασίας είναι οι εξής:

1. Σύνδεση των χρηστών. Οι χρήστες θα έχουν την δυνατότητα να συνδέονται (login) με ένα μοναδικό αναγνωριστικό και να αποθηκεύουν τα δεδομένα τους στο cloud.
2. Εξατομίκευση των προπονητικών προγραμμάτων: Η εφαρμογή θα παρέχει τη δυνατότητα εξατομίκευσης των προπονητικών προγραμμάτων, επιτρέποντας στους χρήστες να προσαρμόζουν τις προπονήσεις τους στις ατομικές τους ανάγκες και στόχους.
3. Παρακολούθηση της προόδου: Οι χρήστες θα έχουν τη δυνατότητα να παρακολουθούν την πρόοδό τους, καταγράφοντας τα βάρη, τις επαναλήψεις, και τον χρόνο που δαπανούν σε κάθε άσκηση.
4. Επεξεργασία προσωποποιημένων δεδομένων. Οι χρήστες θα έχουν την δυνατότητα να επεξεργάζονται ορισμένα προσωποποιημένα στοιχεία τους, όπως εικόνες, σημειώσεις κ.α.

Ένας έμμεσος στόχος είναι η προώθηση της φυσικής κατάστασης και της υγείας των χρηστών μέσω της παροχής ενός εργαλείου που θα τους εμπνέει να είναι πιο δραστήριοι και υγιείς.

Αυτοί οι στόχοι αντικατοπτρίζουν τη φιλοδοξία της εργασίας να δημιουργήσει μια εφαρμογή που όχι μόνο θα προσφέρει λειτουργικότητα, αλλά θα ενισχύσει επίσης την υγεία και την ευεξία των χρηστών μέσω της παρακολούθησης και της βελτίωσης της φυσικής τους κατάστασης

# **Δομικά στοιχεία της εφαρμογής**

Η επιτυχία μιας κινητής εφαρμογής βασίζεται σε μεγάλο βαθμό στα δομικά της στοιχεία, τα οποία καθορίζουν την απόδοση, τη λειτουργικότητα και την αξιοπιστία της. Στο πλαίσιο αυτής της εργασίας, θα εξετάσουμε εκτενώς τα δομικά στοιχεία της εφαρμογής μας, η οποία αναπτύχθηκε χρησιμοποιώντας τεχνολογίες όπως η Dart, το Flutter και το Firebase.

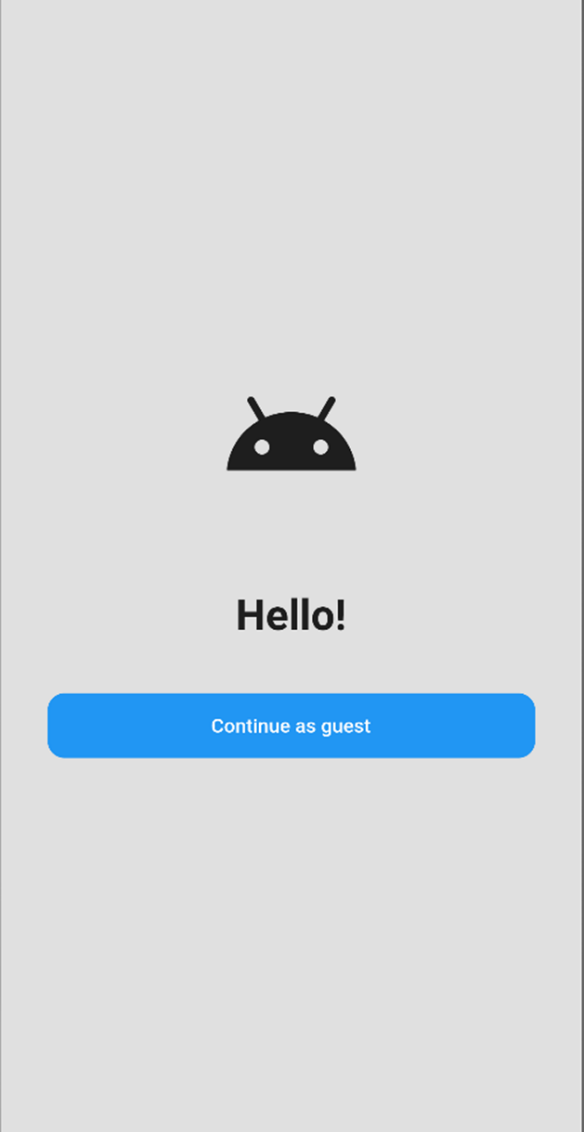
## Η γλώσσα προγραμματισμού Dart

# **Περιγραφή της Εφαρμογής**

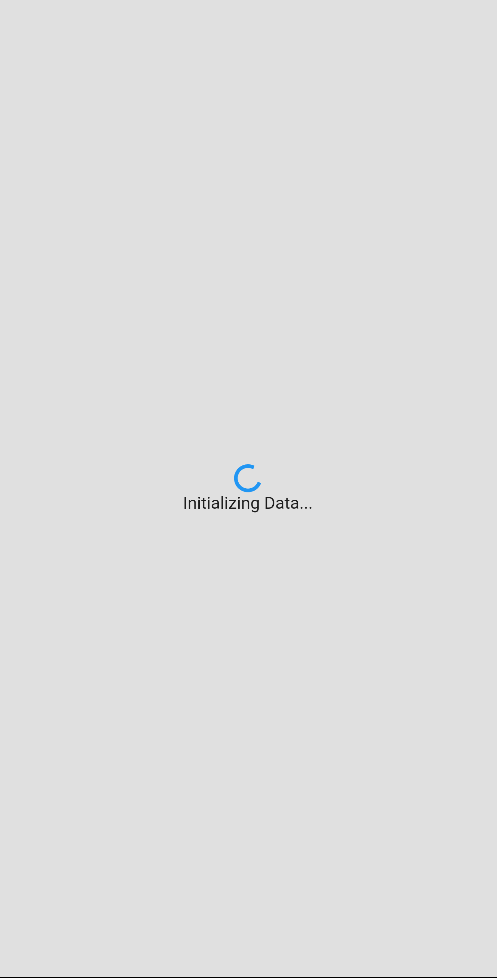
Σε αυτό το κεφάλαιο, θα εξετάσουμε αναλυτικά τη λειτουργικότητα της εφαρμογής μας, του "Workout Tracker." Θα εξηγήσουμε πώς οι χρήστες μπορούν να αλληλεπιδρούν με την εφαρμογή, ποιες λειτουργίες διατίθενται και πώς αυτές οδηγούν σε μια ομαλή και αποτελεσματική εμπειρία. Θα δώσουμε έμφαση στην προσαρμογή, τη δημιουργία προπονητικών προγραμμάτων, την καταγραφή προόδου και τη διαχείριση του προσωπικού προφίλ.

## Είσοδος στην εφαρμογή

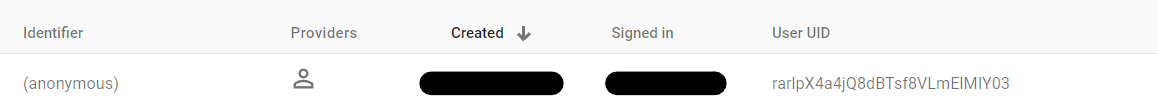
Η πρώτη επαφή του χρήστη με την εφαρμογή είναι η διαδικασία εισόδου. Μόλις ο χρήστης πατήσει “Continue as a guest” τότε δημιουργείται ένας ανώνυμος χρήστης με ένα τυχαίο όνομα χρήστη στο google Firebase και στη συνέχεια λαμβάνει μέρος μιας αρχικοποίηση, η οποία παράγει αυτόματα στην βάση δεδομένων στο google Firebase, ένα σύνολο βασικών ασκήσεων και κάποια έτοιμα προπονητικά προγράμματα.



Εικόνα 1 - Σελίδα σύνδεσης.

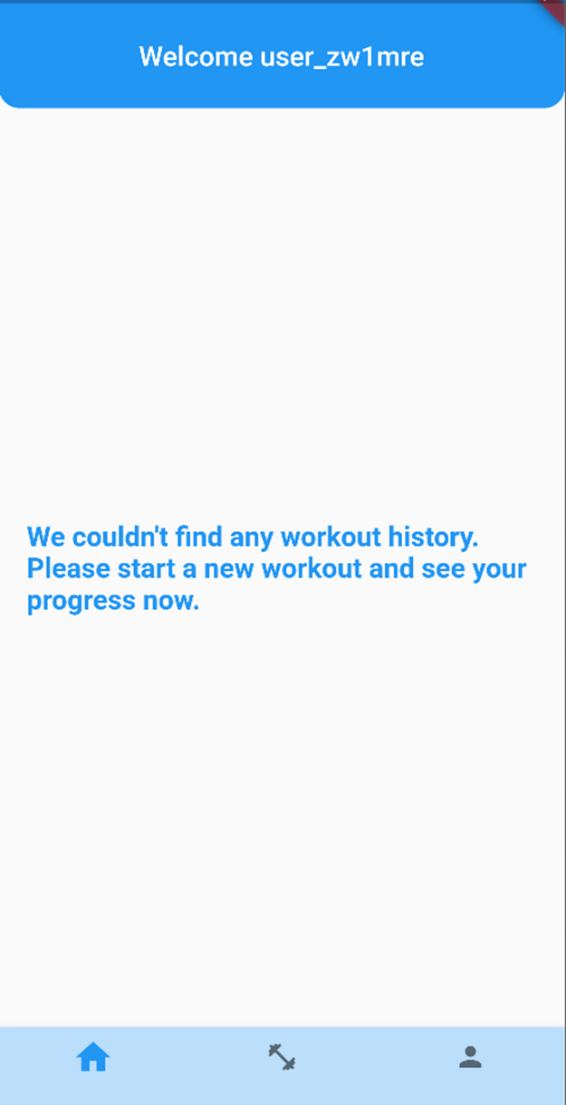


Εικόνα 2 - Αρχικοποίηση δεδομένων.



Εικόνα 3 - Εγγραφή βάσης δεδομένων.

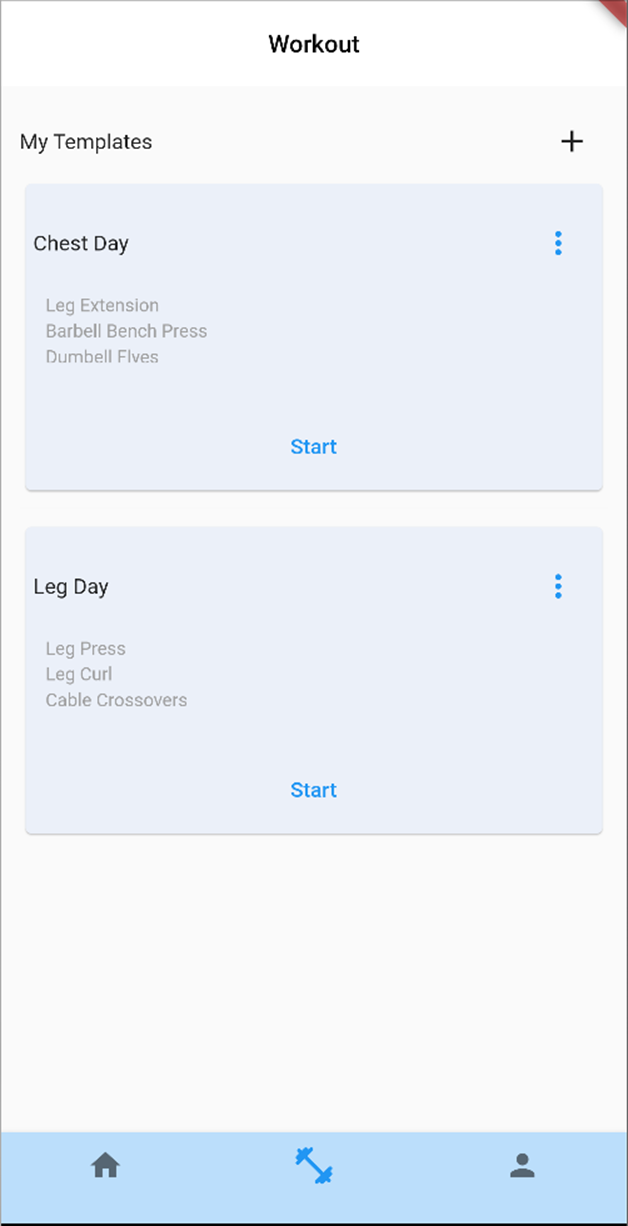
Μετά την είσοδο στην εφαρμογή ο χρήστης έρχεται σε επαφή με το ιστορικό καταγραφής προόδου το οποίο αρχικά είναι άδειο καθώς δεν έχει προχωρήσει στην χρήση κάποιου προπονητικού προγράμματος.



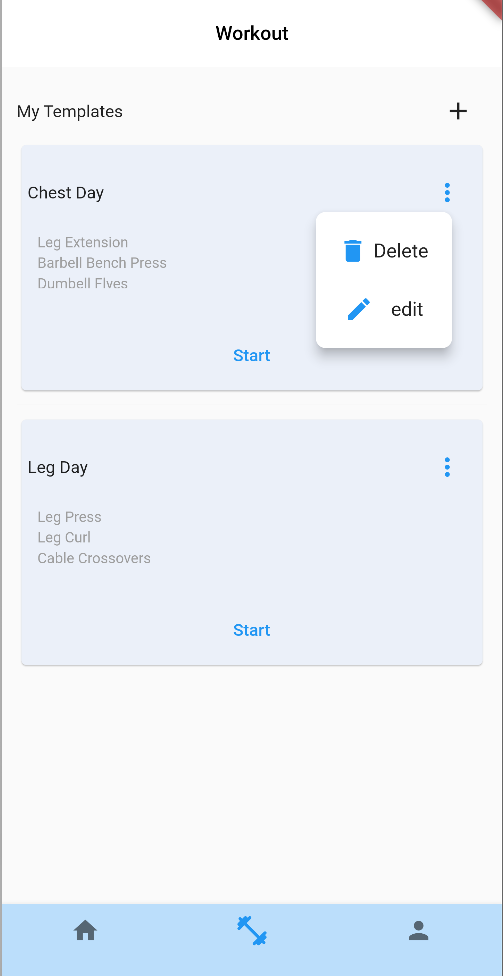
Εικόνα 4 - Κενό ιστορικό προπονήσεων.

## Προβολή λίστας προπονητικών προγραμμάτων/προπονήσεων

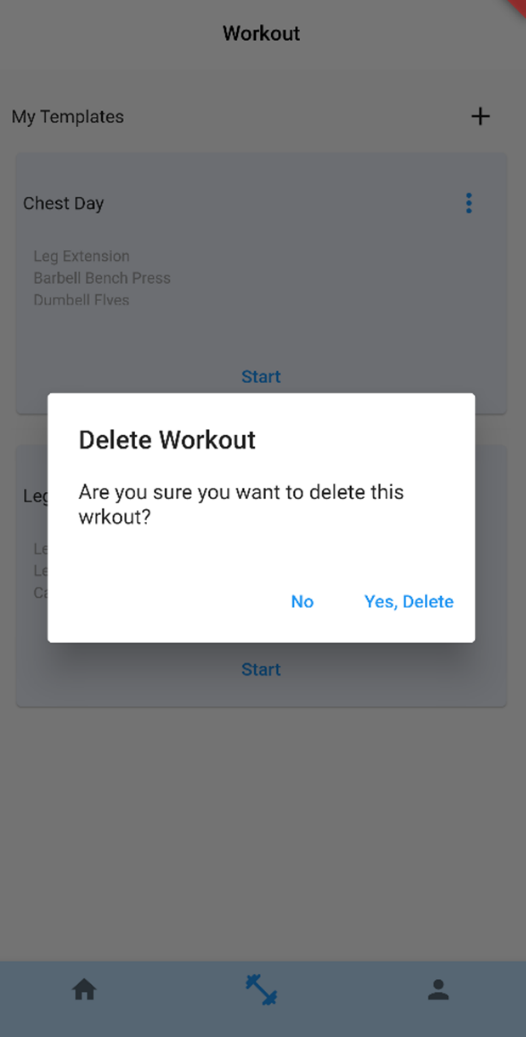
Κύριο στοιχείο της εφαρμογής είναι η δυνατότητα δημιουργίας προσαρμοσμένων προπονητικών προγραμμάτων. Μεταβαίνοντας το δεύτερο εικονίδιο της εφαρμογής ο χρήστης έρχεται σε επαφή με την λίστα των έτοιμων προπονητικών προγραμμάτων (που δημιουργήθηκαν κατά τη πρώτη σύνδεση του χρήστη), τα οποία ο χρήστης έχει τη δυνατότητα να επεξεργαστεί να διαγράψει. Επίσης έχει την δυνατότητα δημιουργήσει κάποιο νέο προπονητικό πρόγραμμα, καθώς και να πραγματοποιήσει κάποια προπόνηση.



Εικόνα 5 - Λίστα προπονήσεων.



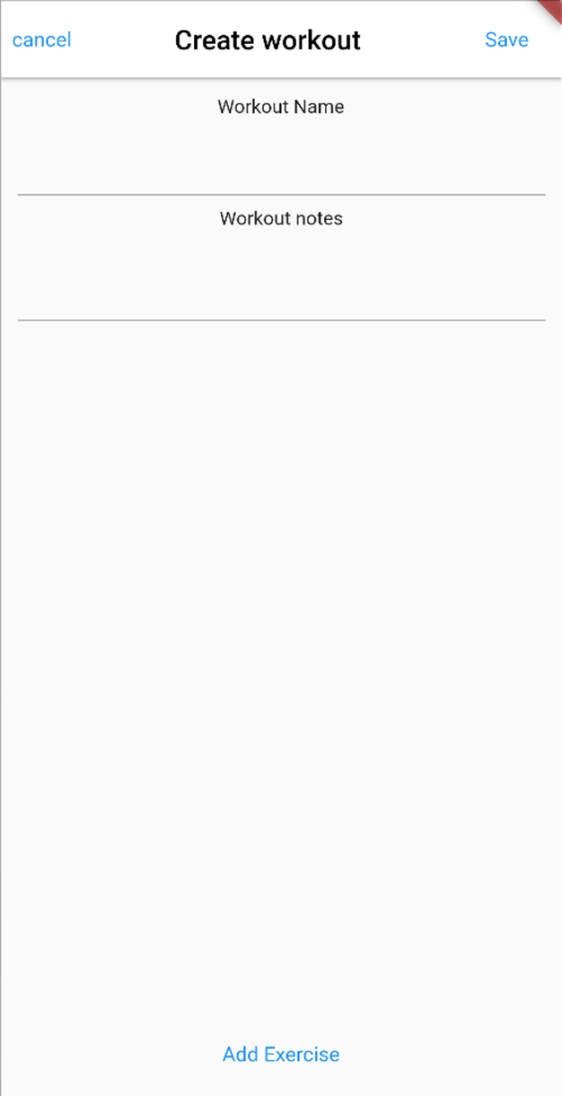
Εικόνα 6 - Επιπλέον επιλογές προπόνησης.



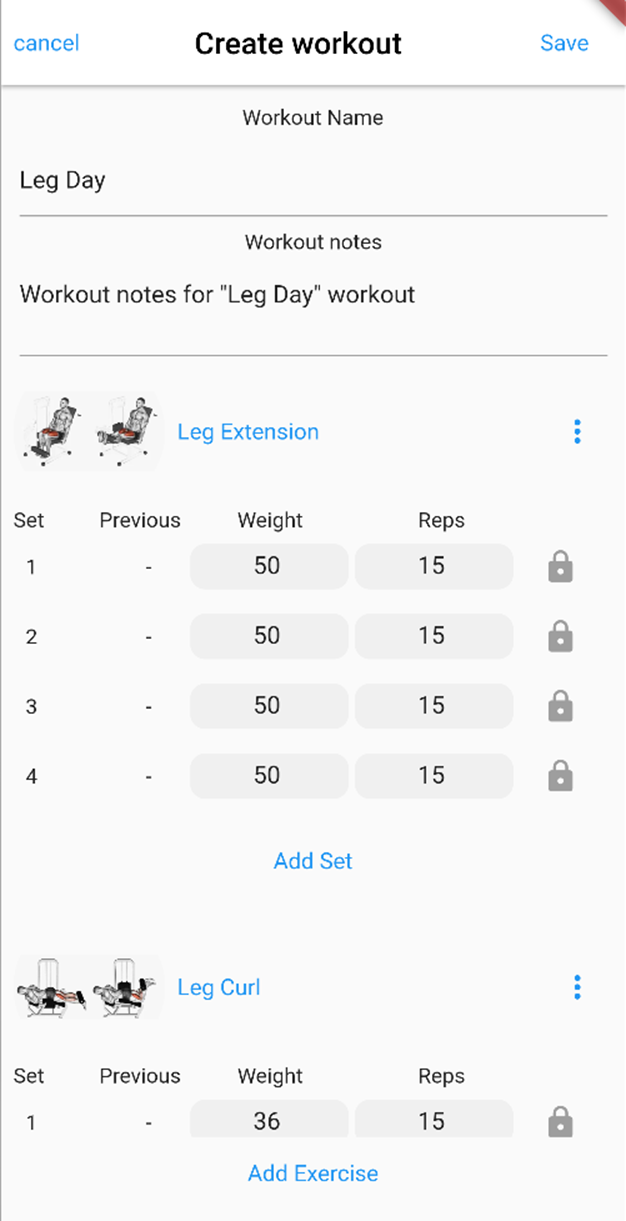
Εικόνα 7 - Μήνυμα διαγραφής προπόνησης.

## Δημιουργία/ επεξεργασία προπονητικού προγράμματος

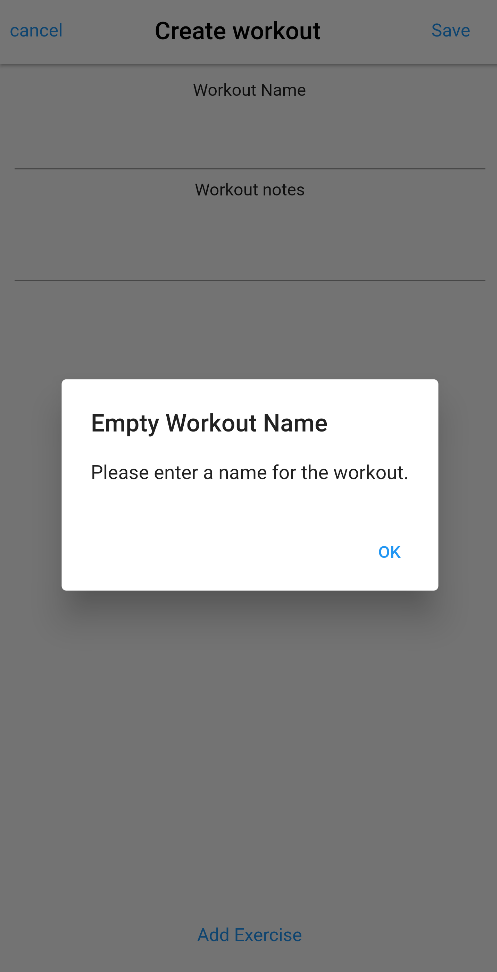
Ο χρήστης της εφαρμογής, κατά την δημιουργία και επεξεργασία κάποιου προπονητικού προγράμματος μπορεί να προσθέσει όνομα, κάποιες σημειώσεις καθώς και την λίστα ασκήσεων που θα πραγματοποιήσει κατά την διάρκεια την προπόνησης. Κάθε άσκηση, περιέχει έναν αριθμό σετ τα οποία περιέχουν τον σύνολο των βαρών σε κιλά, το σύνολο επαναλήψεων τον αριθμό του σετ καθώς και το σύνολο των προηγουμένων σετ (κιλά x επαναλήψεις) Αν δεν έχει εκτελέσει την συγκεκριμένη προπόνηση στο παρελθόν, το πεδίο είναι “-“. Ο χρήστης μπορεί να προσθέσει ασκήσεις πατώντας το κουμπί “Add exercise”.



Εικόνα 8 - Δημιουργία νέας προπόνησης.



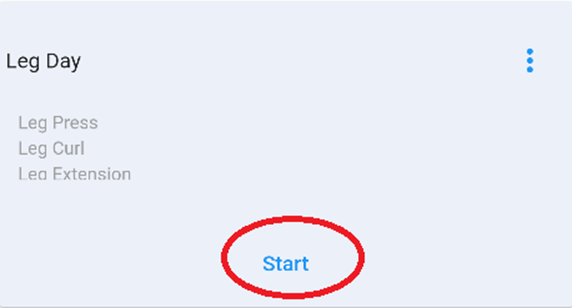
Εικόνα 9 - Προπόνηση με συμπληρωμένα πεδία.



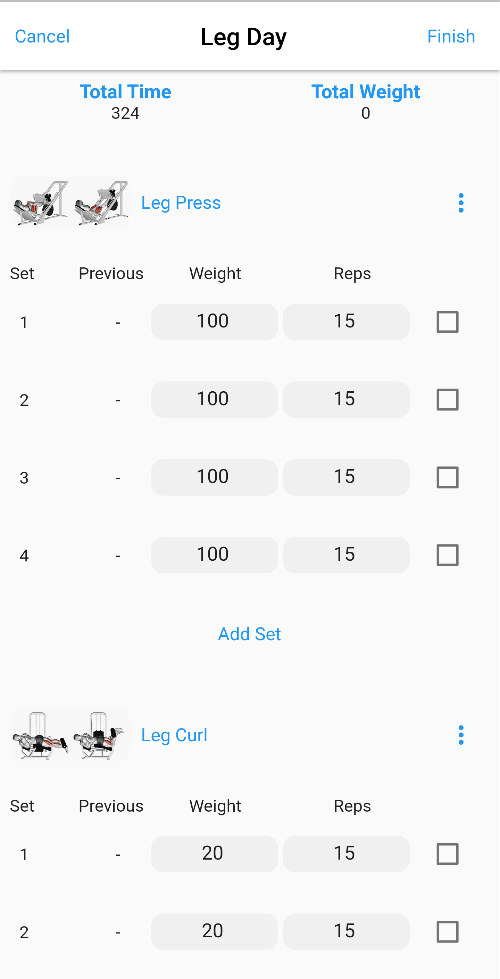
Εικόνα 10 - Μήνυμα σφάλματος.

## Πραγματοποίηση προπόνησης.

Μια προπόνηση μπορεί να πραγματοποιηθεί πατώντας την επιλογή “Start” στο προπονητικό πρόγραμμα της επιλογής του χρήστη.

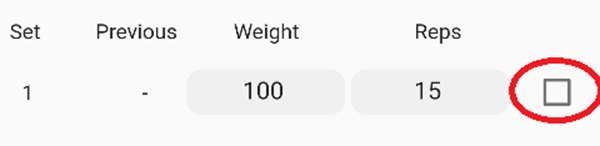


Εικόνα 11 - Έναρξη προπόνησης.



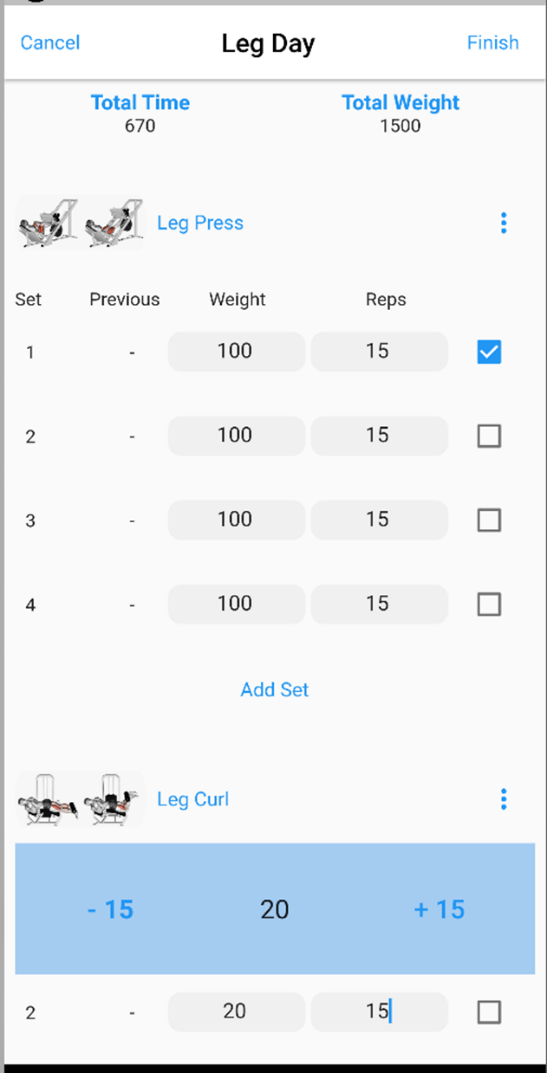
Εικόνα 12 - Σελίδα πραγματοποίησης προπόνησης.

Κατά την διάρκεια της προπόνησης, ο χρήστης μπορεί να επεξεργαστεί τα βάρη και τις επαναλήψεις των σετ, μπορεί να προσθέσει και να αφαιρέσει ασκήσεις. Επίσης, μια διαφορά από την σελίδα της δημιουργίας/επεξεργασίας είναι ότι ο χρήστης, μπορεί να δηλώσει μια άσκηση ως ολοκληρωμένη.



Εικόνα 13 - Δήλωση άσκησης ως ολοκληρωμένη.

Αυτό έχει ως αποτέλεσμα να εμφανιστεί ένας μετρητής στο κάτω μέρος, ο οποίος λειτουργεί ως χρονόμετρο για αποθεραπεία μέχρι το επόμενο σετ.



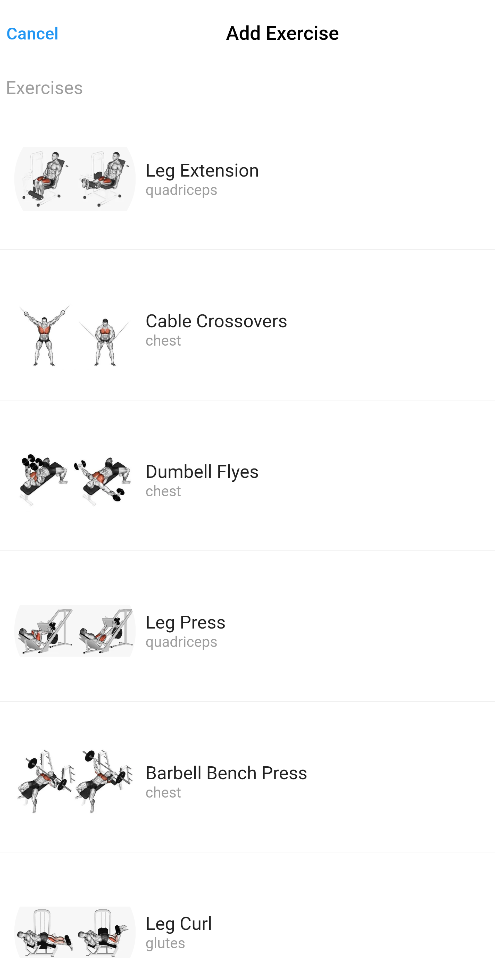
Εικόνα 14 - Χρονόμετρο αποθεραπείας.

Τέλος, στο πάνω μέρος της σελίδας ο χρήστης μπορεί να δει τον συνολικό χρόνο της προπόνησης, και το σύνολο κιλών των ασκήσεων (κιλά \* επαναλήψεις).

Πατώντας το κουμπί “Finish”, η προπόνηση αποθηκεύεται στην Google Firebase βάση δεδομένων και δημιουργείται μία εγγραφή της προπόνησης στο ιστορικό καταγραφής.

## Λίστα ασκήσεων

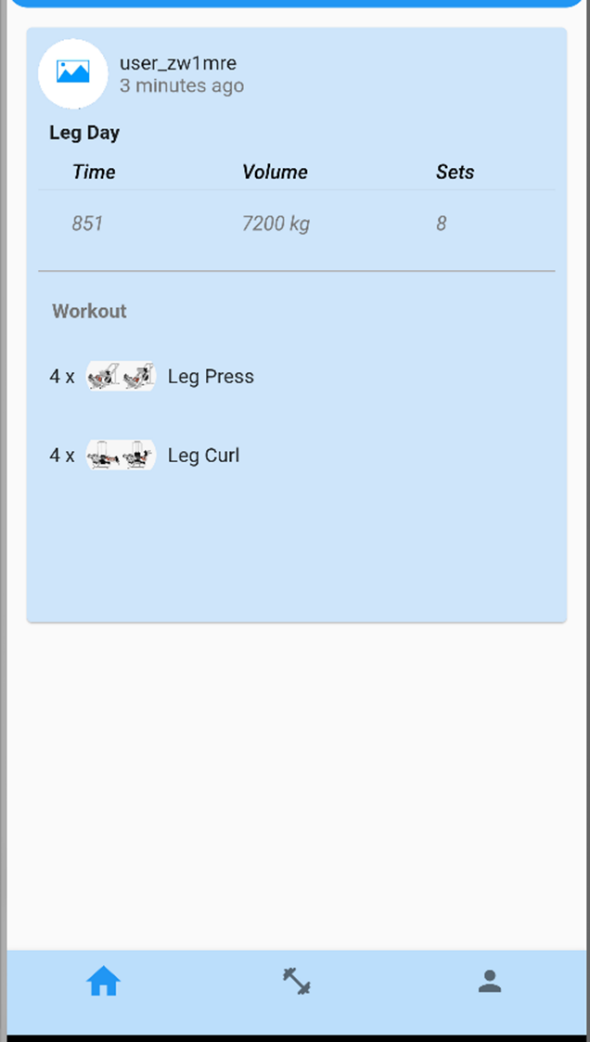
Ο χρήστης, στην οθόνη της προπόνησης (προσθήκη, επεξεργασία, πραγματοποίηση) έχει την επιλογή να προσθέσει ασκήσεις. Πατώντας το κουμπί “Add exercise” εμφανίζεται μια λίστα με ασκήσεις που δημιουργήθηκαν κατά την πρώτη σύνδεση του χρήστη και πατώντας πάνω στην άσκηση, αυτή προστίθεται στην προπόνηση.



Εικόνα 15 - Λίστα ασκήσεων.

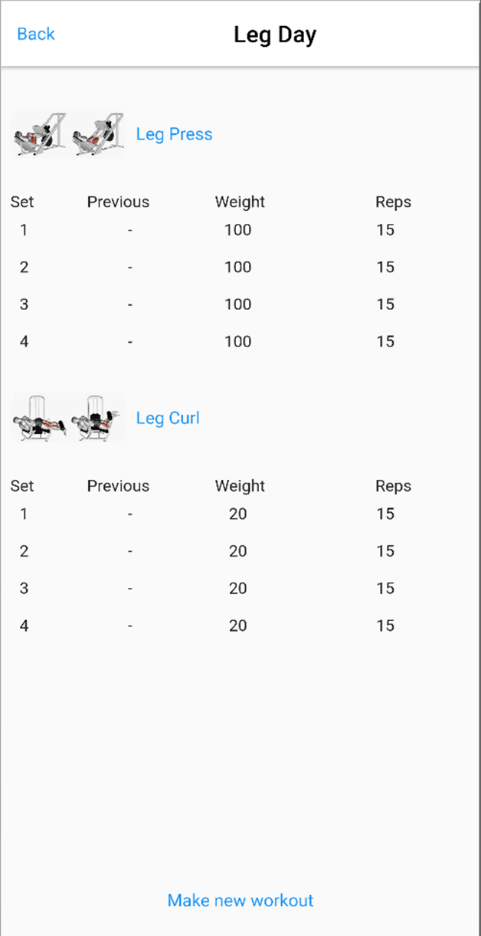
## Διαχείριση του Ιστορικού

Οι χρήστες έχουν πρόσβαση στο ιστορικό των προπονητικών προγραμμάτων που έχουν ακολουθήσει στο παρελθόν.



Εικόνα 16 - Ιστορικό προπονήσεων με εγγραφές.

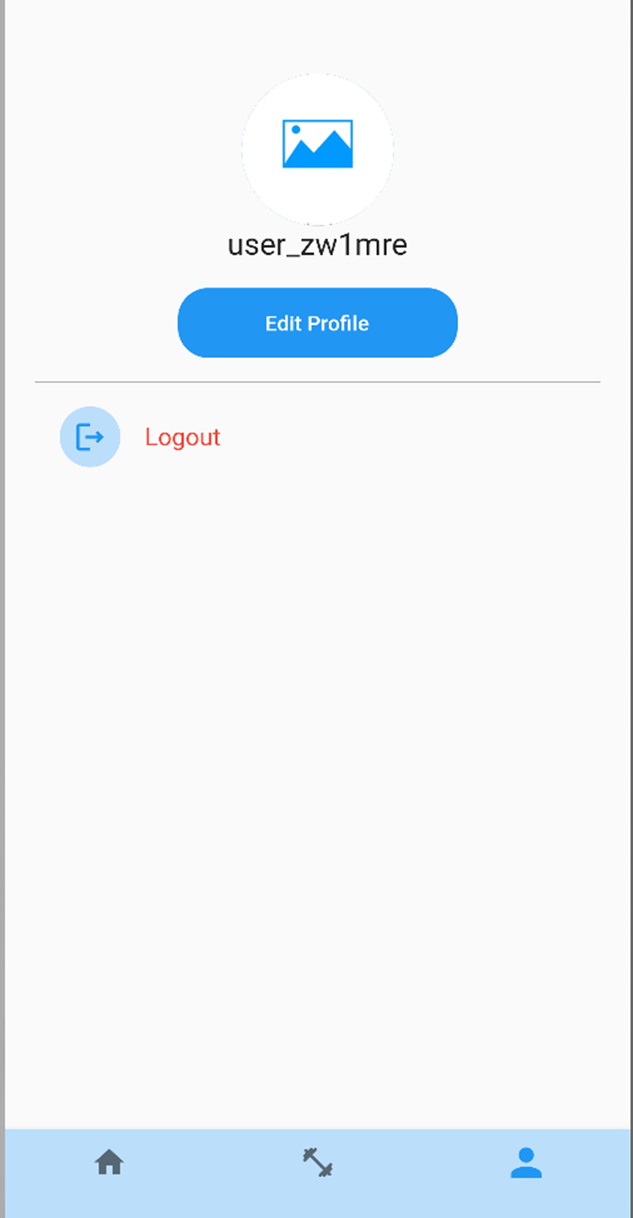
Ένας χρήστης εκτός από την απλή παρακολούθηση του ιστορικού, έχει και την επιλογή να δει περισσότερες πληροφορίες για την προπόνηση πατώντας πάνω στο τετράγωνο της επιλογής του, όπως επίσης να δημιουργήσει νέο προπονητικό πρόγραμμα βασισμένο σε αυτό πατώντας το “Make new workout”.



Εικόνα 17 - Επιπλέον πληροφορίες από το ιστορικό προπονήσεων.

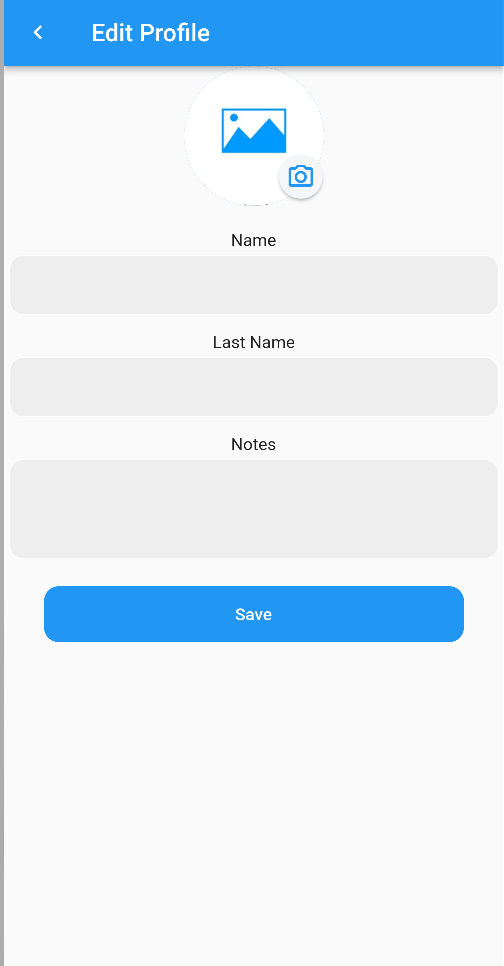
## Επεξεργασία Προσωπικού Προφίλ

Πατώντας στο τρίτο εικονίδιο, ο χρήστης έρχεται σε επαφή με την οθόνη που περιέχει πληροφορίες για αυτόν καθώς και την επιλογή να αποσυνδεθεί.



Εικόνα 18 - Σελίδα χρήστη.

Πατώντας το κουμπί “Edit profile”, να επεξεργαστεί ορισμένες πληροφορίες όπως το όνομα, το επίθετο του, σημειώσεις καθώς και μπορεί να προσθέσει κάποια εικόνα.



Εικόνα 19 - Σελίδα επεξεργασίας προσωποποιημένων δεδομένων χρήστη.

# **Συμπεράσματα – Μελλοντικές κατευθύνσεις / Conclusions – Future Directions**

# **Βιβλιογραφία / References**

# **Παραρτήματα / Appendixes**

Η εφαρμογή Αυτή είναι γραμμένη με την χρήση του framework flatter της Google και για γλώσσα προγραμματισμού έχει χρησιμοποιηθεί η γλώσσα dart

Η Dart είναι ένα an open source έργο που ξεκίνησε και συνεχώς αναπτύσεται από τη Google, επωφελούμενο από μια ζωντανή και ενεργή κοινότητα χρηστών που συμβάλλουν ενεργά στην ανάπτυξή του και διατηρούν έναν μεγάλο αποθετήριο βιβλιοθηκών και πακέτων που είναι διαθέσιμα στη διεύθυνση <https://pub.dev/>.

Dart is a type-safe garbage collected, client-optimized OOP language for developing fast apps on any platform. Dart is designed for a technical envelope that is particularly suited to client development, prioritizing both development (sub-second stateful hot reload) and high-quality production experiences across a wide variety of compilation targets (web, mobile, and desktop).

Dart also forms the foundation of Flutter. Dart provides the language and runtimes that power Flutter apps, but Dart also supports many core developer tasks like formatting, analyzing, and testing code.----na paei sto flutter auto

Dart’s compiler technology lets you run code in different ways:

* **Native platform**: For apps targeting mobile and desktop devices, Dart includes both a Dart VM (provides an execution environment for the dart programming language) with just-in-time (JIT) compilation that compiles the necessary code while the program is executing, making the debugging experience better and an ahead-of-time (AOT) compilation that translates dart code to native ARM or x64 machine code.
* **Web platform**: For apps targeting the web, Dart can compile for development or production purposes.

**Flutter**Flutter is an open-source UI software development kit created by Google in 2017. It is used to develop cross-platform native applications for mobile, web, desktop, and embedded devices from a single codebase. Flutter is built using the Dart programming language, which is also developed by Google.

Flutter provides a range of features such as hot reload, widget libraries, material design, Cupertino widgets, animation, and more. It allows developers to build beautiful, natively compiled applications for mobile, web, and desktop with one programming language and single codebase. Flutter has a growing community of developers and contributors who provide support and resources for the framework.

**Firebase**

Firebase is a mobile and web application development platform that was developed by Firebase, Inc. and later acquired by Google in 2014. It provides developers with a set of tools and services to build and manage apps more easily and efficiently. Firebase offers various features and services to help developers with tasks such as app development, database management, authentication, cloud storage, and more. Some of the key components and features of Firebase include:

1. Realtime Database: Firebase provides a NoSQL, cloud-hosted database that allows developers to store and synchronize data in real-time across multiple clients. This is particularly useful for building collaborative and interactive applications.
2. Authentication: Firebase offers robust authentication services, including email and password authentication, social media sign-in (e.g., Google, Facebook, Twitter), and anonymous authentication. This simplifies user management and ensures secure access to your app.
3. Hosting: Firebase Hosting allows developers to deploy web apps quickly and securely. It provides a content delivery network (CDN) for fast global hosting and serves as a platform for hosting static and dynamic content.
4. Cloud Functions: Firebase Cloud Functions lets you run serverless functions in response to Firebase events or HTTP requests. This enables you to extend and customize your app's behavior on the server side.
5. Cloud Firestore: Firestore is Firebase's next-generation document-based NoSQL database, offering more advanced querying and data modeling capabilities compared to the Realtime Database.
6. Cloud Storage: Firebase Storage provides scalable cloud-based storage for user-generated content, such as images, videos, and other files.
7. Cloud Messaging: Firebase Cloud Messaging (FCM) allows you to send push notifications to users on Android, iOS, and web platforms, helping you engage and retain your audience.
8. Analytics: Firebase Analytics provides insights into user behavior and app performance, helping you make data-driven decisions to improve your app.
9. Performance Monitoring: Firebase Performance Monitoring helps you track and optimize your app's performance, including app startup times, network requests, and more.
10. Remote Config: Firebase Remote Config enables you to remotely change the behavior and appearance of your app without publishing app updates.
11. A/B Testing: Firebase A/B Testing allows you to create experiments to test different variations of your app to see which ones perform better with users.
12. Authentication UI: Firebase offers pre-built UI components for common authentication flows, making it easier to create a consistent and user-friendly authentication experience.

Firebase is a versatile platform suitable for a wide range of applications, from small personal projects to large-scale enterprise solutions. Its integration with other Google Cloud services and its ease of use make it a popular choice among developers for building and managing modern web and mobile apps.

**+++ firebase offers a helpful suite of tools that make integration with flutter apps easier.**[**https://firebase.flutter.dev/**](https://firebase.flutter.dev/)

**State management**

e State management is a crucial aspect of software development, particularly in the context of user interface (UI) programming. It refers to the handling of the data that a program needs to keep track of while it’s running.

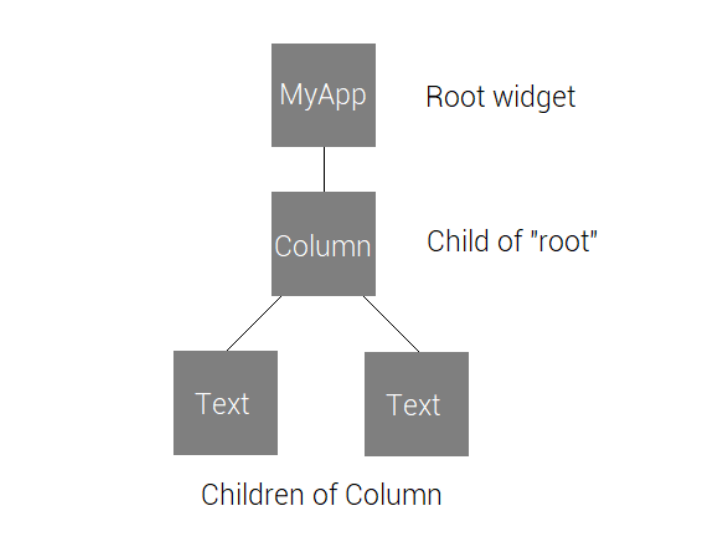
In the context of UI programming, state management involves tracking variables that can influence the output of a render method over time. This could include user inputs, configuration settings, or data fetched from a database.

For example, consider a simple counter app. The current count is part of the app’s state. When the user presses a button to increment the count, the state changes and the UI updates to reflect this new state.

Effective state management ensures that an application behaves predictably and is easier to test and debug. It can also improve performance by preventing unnecessary computations or renderings.

**Widgets**

In Flutter everything that appears on the screen is called "widget" because, technically speaking, it’s a descendant of the Widget class. When you create user interfaces in Flutter you make a composition of widgets by nesting them one inside the other. When you nest widgets one inside the other you create a hierarchy called "widget tree" in which there are parents and children.



The addition of widgets makes the tree bigger and places new items on the screen. The context parameter in build(BuildContext context) gives important information about the position of the leaf in the tree. In particular:

The method Widget build(BuildContext context) adds new leaves to the widget tree in order to place new graphical items in the UI. Widgets are nested one inside the other using named parameters in the constructors to make the reading of the code very expressive

• A BuildContext instance is used by Flutter to know details about the widget when the tree is being traversed;

• We will see that calling SomeWidget.of(context) returns the closest widget in the tree whose type is SomeWidget;

• Each widget has its own BuildContext instance which becomes the parent context of the widget(s) returned by its build method.

A class becomes a Flutter widget when it subclasses StatelessWidget or StatefulWidget and overrides the Widget build(...); abstract method. That’s it: the main task of a widget is laying out other widgets on the tree using the build() method.

Before creating a widget the developer must decide whether the state will change during the time or not. If the state changes at some point, then it means that something has happened such as:

* the user has tapped on a button and thus something in the UI must change;
* the device has been rotated and the UI must be repainted;
* there’s a new event on a stream and a widget depending on it is notified (and thus a rebuild happens in order to reflect the changes brought by the stream).

**Stateless widget**. Use this kind of widget when you need to create a piece of UI that is not going to change over the time. It’s a "standalone" block that doesn’t depend on external events or sources; it just relies on its constructor and the internal data

Stateful widget. Use this kind of widget when you need to create a piece of UI that is going to change over the time. In this case the UI is going to dynamically change due to external events such as the received response of an HTTP request or the callback triggered by a button tap.

**State management** in Flutter is a complex topic. It is one of the most important processes in the life cycle of an application. In Flutter, state is whatever data you need in order to rebuild your UI at any moment in time. When this data changes, it triggers a redraw of the user interface2.

There are several state management strategies in Flutter, including setState, Provider, BLoC, GetX, and Riverpod.

setState: This is the low-level approach to use for widget-specific, ephemeral state. It’s a function called inside a StatefulWidget to modify its state and then rebuild it.

Provider: This is a wrapper around InheritedWidget to make them easier to use and more reusable4. It’s easy to understand and doesn’t use a lot of code4.

BLoC: Business Logic Component (BLoC) is a state management system for Flutter recommended by Google developers. It separates the business logic from UI4.

GetX: It’s a state management solution that combines utility functions with a simple syntax for developers.

Riverpod: This works in a similar fashion to Provider. It offers compile safety and testing without depending on the Flutter SDK3.

Choosing between these strategies depends on your specific needs and familiarity with these concepts. However, if you’re a beginner in Flutter and don’t have a solid reason to choose another approach, Provider might be the best place to start4. Here’s why:

Simplicity: Provider is relatively easy to understand and doesn’t require writing a lot of code4.

Flexibility: Provider can be used with other state management techniques like BLoC, making it more flexible4.

Efficiency: Provider ensures your UI is efficient. By using Provider, only the widgets that are listening to a particular piece of data will rebuild when that data changes4.

Community Support: Provider is widely used in the Flutter community, which means you’ll find lots of resources and support.

Remember, there’s no one-size-fits-all solution when it comes to state management. The best approach depends on your app’s needs, complexity, and the development team’s experience.

**Περιγραφή εφαρμογής.**

**Κεφάλαιο [Αριθμός Κεφαλαίου]: Παρουσίαση της Λειτουργικότητας**

**Εισαγωγή**

Συμπέρασμα  
Στο πλαίσιο αυτής της μελέτης, διερευνήσαμε τη δημιουργία και τη λειτουργία της εφαρμογής "Workout Tracker" σε περιβάλλον κινητής τεχνολογίας. Η εφαρμογή αυτή αναπτύχθηκε με στόχο να παρέχει ένα αξιόπιστο και εύχρηστο εργαλείο για την καταγραφή, την παρακολούθηση και τη βελτίωση της φυσικής κατάστασης των χρηστών.

Κατά τη διάρκεια της ανάπτυξης της εφαρμογής, επικεντρωθήκαμε στη δημιουργία μιας ευέλικτης πλατφόρμας που επιτρέπει στους χρήστες να προσαρμόζουν τα προπονητικά τους προγράμματα, να παρακολουθούν την πρόοδό τους και να διαχειρίζονται το προσωπικό τους προφίλ.

Μέσω αυτής της εφαρμογής, οι χρήστες έχουν την ευκαιρία να βελτιώσουν τη φυσική τους κατάσταση, να αυξήσουν τη σωματική τους δύναμη και αντοχή, και να επιτύχουν τους προσωπικούς τους στόχους για έναν υγιή τρόπο ζωής.

Τέλος, αναγνωρίζουμε ότι η επιτυχία της εφαρμογής "Workout Tracker" εξαρτάται από τη συνεχή υποστήριξη και ανάδειξη των αναγκών των χρηστών μας. Μελλοντικές αναβαθμίσεις και επεκτάσεις στη λειτουργικότητα της εφαρμογής θα πρέπει να λαμβάνουν υπόψη τις ανάγκες και τα αιτήματα των χρηστών προκειμένου να διατηρήσουμε μια ισχυρή και ανταγωνιστική παρουσία στον χώρο των εφαρμογών υγείας και φυσικής κατάστασης.

**------------------------------------**

Ο χρήστης θα έχει την ευκαιρία να συνδέεται αυτόματα στην εφαρμογή, δίνοντάς του την δυνατότητα να δημιουργεί προσαρμοσμένα προπονητικά προγράμματα, ανάλογα με τις δικές του προτιμήσεις και ανάγκες. Αυτό σημαίνει ότι ο χρήστης μπορεί να καταρτίζει δικές του ασκήσεις, να δημιουργεί πλήρη προπονητικά προγράμματα, και να προσαρμόζει το περιεχόμενο ανάλογα με τους στόχους του.

Επιπλέον, η εφαρμογή θα προσφέρει στον χρήστη την δυνατότητα να διατηρεί ένα ιστορικό των προπονητικών προγραμμάτων που έχει ακολουθήσει στο παρελθόν, παρέχοντας τη δυνατότητα να ανατρέχει στην πρόοδό του. Αυτό θα επιτρέπει στον χρήστη να παρακολουθεί την εξέλιξη της φυσικής του κατάστασης, καταγράφοντας βάρη, επαναλήψεις και χρόνο που αφιερώνει σε κάθε άσκηση.

Τέλος, ο χρήστης θα έχει τη δυνατότητα να προσαρμόζει το προφίλ του στις ατομικές του ανάγκες και προτιμήσεις, προσθέτοντας πληροφορίες σχετικά με την ηλικία, το βάρος, το ύψος, το επίπεδο φυσικής δραστηριότητας, και τους στόχους του για τη βελτίωση της φυσικής του κατάστασης. Αυτή η εξατομίκευση θα βοηθά την εφαρμογή να προτείνει πιο αποτελεσματικά προπονητικά προγράμματα και άσκησες, προσαρμοσμένα στις ανάγκες και τους στόχους του χρήστη.