

Εργαλεία:

_____ Για την υλοποίηση της εργασίας χρησιμοποιήθηκε το IDE “Android Studio” και η γλώσσα προγραμματισμού Java.

Ανάλυση:

_____ Τα περιεχόμενα της εργασίας χωρίζονται στις εξής κατηγορίες:

- Κλάσεις Java , όπου περιέχεται ο κώδικας χειρισμού της λογικής της εφαρμογής.
- Αρχεία XML, στα οποία περιέχονται οι πληροφορίες για την δόμηση των διαφόρων Activities, που χρησιμοποιούνται.

Κάθε αρχείο XML, περιέχει και την συνοδευτική του Java κλάση, ενώ υπάρχουν επιπλέον Java κλάσεις για τη δημιουργία και τον χειρισμό μιας βάσης δεδομένων σταθερής αποθήκευσης δεδομένων. Από τις κλάσεις αυτές αρχίζει και η ανάλυση.

Βάση δεδομένων:

_____ Η βάση δεδομένων που στήθηκε για την εξυπηρέτηση των αναγκών της εφαρμογής περιγράφεται μέσω δύο κλάσεων: την κλάση User.java και την κλάση UserDbManageClass.

Κλάση User.java:

Η συγκεκριμένη κλάση χρησιμοποιείται για την περιγραφή της οντότητας της βάσης δεδομένων. Συγκεκριμένα υπάρχει μία μοναδική οντότητα που ονομάζεται “user” και έχει τα παρακάτω χαρακτηριστικά:

- Private int id: Αποτελεί το κύριο κλειδί του πίνακα της βάσης και είναι μοναδικό για κάθε εγγραφή.
- Private String name: Το όνοματεπώνυμο του χρήστη
- Private String height: Το ύψος του χρήστη (σε εκατοστά)
- Private String weight: Η μάζα του χρήστη (σε χιλιόγραμμα) Private String BMI: Ο δείκτης μάζας σώματος.

Επιπλέον από μεθόδους υπάρχουν οι εξής:

- `Public User(int id, String name, String height, String weight, String BMI)`: Η συγκεκριμένη μέθοδος αποτελεί κατασκευαστής της κλάσης και αρχικοποιεί ένα αντικείμενο `User` με τα ορίσματα που δέχεται σαν είσοδο.
- `Public User(String name, String height, String weight, String BMI)`: Η συγκεκριμένη μέθοδος αποτελεί κατασκευαστής της κλάσης και αρχικοποιεί ένα αντικείμενο `User` με τα ορίσματα που δέχεται σαν είσοδο. Μόνη διαφορά με την παραπάνω μέθοδο είναι ότι δεν αρχικοποιείται το `id`.
- `Public User()`: Η μέθοδος αυτή είναι ο κενός κατασκευαστής της κλάσης. `Setters`, `Getters` για όλα τα χαρακτηριστικά του της κλάσης.

Κλάση `UserDbManageClass.java`:

Η συγκεκριμένη κλάση αποτελεί κλάση χειρισμού της βάσης δεδομένων που δημιουργείται για τις ανάγκες της εφαρμογής. Αυτή κληρονομεί ιδιότητες και μεθόδους της κλάσης `SQLiteOpenHelper`. Η κλάση αυτή περιέχει τις εξής ιδιότητες:

- `public static final int DATABASE_VERSION`: Ένας ακέραιος αριθμός που αναπαριστά την “έκδοση” της βάσης
- `public static final String DATABASE_NAME`: Ένα `String` που περιέχει το όνομα της βάσης
- `public static final String TABLE_USERS`: Ένα `String` που περιέχει το όνομα του πίνακα της βάσης.
- `public static final String COLUMN_ID`: Ένα `String` που περιέχει το όνομα της στήλης `id`.
- `public static final String COLUMN_NAME`: Ένα `String` που περιέχει το όνομα της στήλης `name`.
- `public static final String COLUMN_HEIGHT`: Ένα `String` που περιέχει το όνομα της στήλης `height`.
- `public static final String COLUMN_WEIGHT`: Ένα `String` που περιέχει το όνομα της στήλης `weight`.
- `public static final String COLUMN_BMI`: Ένα `String` που περιέχει το όνομα της στήλης `bmi`.
- `public static final String COLUMN_DATE`: Ένα `String` που αποθηκεύει την ημερομηνία που έγινε η καταχώρηση.

- `public static final String COLUMN_BMIIMG:`
Όλα τα παραπάνω χρησιμοποιούνται σαν σταθερά Strings για την διατύπωση ερωτημάτων SQL πάνω στην βάση δεδομένων.

Ακόμα, η κλάση περιέχει και τις παρακάτω μεθόδους:

- `public UserDbManageClass(Context context, String name, SQLiteDatabase.CursorFactory factory, int version):` Αποτελεί κατασκευαστή της κλάσης και χρησιμοποιεί τον κατασκευαστή της κλάσης γονέα.
- `public void onCreate(SQLiteDatabase db):` Μέθοδος που γίνεται override απ την κλάση γονέα. Εκτελείται όταν πρώτο - δημιουργείται η βάση δεδομένων. Σ αυτήν σχηματίζουμε ένα SQL ερώτημα το οποίο και δημιουργεί τον πίνακα εγγραφών. Η δημιουργία του πίνακα γίνεται με εφαρμογή της μεθόδου `execSQL` στο αντικείμενο `db` που παίρνει η μέθοδος ως όρισμα.
- `public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion):` Ακόμα μια μέθοδος η οποία κάνει override μέθοδο της κλάσης γονέα. Αυτή βέβαια σε αντίθεση με την `onCreate` χρησιμοποιείται, όταν η βάση δεδομένων έχει ήδη δημιουργηθεί σε προηγούμενη εκτέλεση της εφαρμογής και σκοπός είναι η ενημέρωσή της και όχι η δημιουργία της.
- `public void addUser(User user):` Μέθοδος, η οποία υλοποιεί την διαδικασία προσθήκης εγγραφής στον πίνακα της βάσης. Προκειμένου να υλοποιηθεί η λειτουργία αυτή χρησιμοποιούμε ένα αντικείμενο της κλάσης `User` όπου τα πεδία του ταυτίζονται μ' αυτά του πίνακα, αλλά και ένα αντικείμενο της κλάσης `ContentValues` που "κρατάει" τα δεδομένα που θέλουμε να περάσουμε στον πίνακα. Έχοντας όλα τα παραπάνω χρησιμοποιούμε ένα αντικείμενο της κλάσης `SQLiteDatabase` και χρησιμοποιώντας την μέθοδο `insert`.
- `public void findUser(String name):` Μέθοδος, η οποία υλοποιεί την λειτουργία της αναζήτησης εγγραφής με βάση το όνομά του χρήστη (δίνεται ως όρισμα). Ορίζουμε ένα SQL ερώτημα το οποίο εκτελείται με την μέθοδο `rawQuery` που αποθηκεύει τα αποτελέσματά της σε ένα αντικείμενο `Cursor`. Στην συνέχεια επιτελούμε έναν έλεγχο με `if` προκειμένου να ελέγξουμε τι επέστρεψε το ερώτημα SQL. Αν όντως βρέθηκε ο χρήστης

επιστρέφουμε τα δεδομένα του μέσω ενός αντικειμένου User, αλλιώς επιστρέφουμε null.

- `public boolean deleteUser(String name)`: Μέθοδος, η οποία υλοποιεί την λειτουργία της διαγραφής εγγραφής με βάση το όνομα του χρήστη. Ομοίως με την προηγούμενη κάνουμε έλεγχο για την ύπαρξη της εγγραφής στην βάση. Στην περίπτωση που υπάρχει, επιτελείται η διαγραφή χρησιμοποιώντας την μέθοδο `delete` και επιστρέφεται `true`, ενώ αντιθέτως επιστρέφεται `false`, διότι δεν υπάρχει εγγραφή για να διαγραφεί.

Αρχεία περιεχομένου

Κλάση MainActivity.java

Η συγκεκριμένη κλάση αποτελεί την βασική λειτουργία της εφαρμογής (launcher), καθώς είναι το πρώτο Activity που ενεργοποιείται κατά την εκκίνηση της εφαρμογής. Η κλάση κληρονομεί χαρακτηριστικά της κλάσης `AppCompatActivity`. Ο χαρακτήρας της είναι κυρίως προσανατολισμένος στην πλοήγηση μέσα στα υπόλοιπα Activities της εφαρμογής. Περιέχει τις εξής μεθόδους:

- `protected void onCreate(Bundle savedInstanceState)`: μέθοδος, η οποία χρησιμοποιείται κατά την εκτέλεση του Activity και έχει τον ρόλο έναρξης της δραστηριότητας. Αποτελεί override μεθόδου της κλάσης-γονέα και χρησιμοποιεί τον κατασκευαστή της.
- `public void getStarted(View view)`: Μέθοδος `onClick` του κουμπιού "GET STARTED". Χρησιμοποιείται για την δημιουργία ενός Intent για μεταφορά σε άλλο Activity και συγκεκριμένα στο Activity "Calculate" που θα αναλυθεί παρακάτω.
- `public void ToCalendar(View view)`: Μέθοδος `onClick` του κουμπιού "CALENDAR". Χρησιμοποιείται για την δημιουργία ενός Intent για μεταφορά σε άλλο Activity και συγκεκριμένα στο Activity "Calendar" που θα αναλυθεί παρακάτω.
- `public void History(View view)`: Μέθοδος `onClick` του κουμπιού "HISTORY". Χρησιμοποιείται για την δημιουργία ενός Intent για μεταφορά σε άλλο Activity και συγκεκριμένα στο Activity "History" που αναλύεται παρακάτω.

Αρχείο activity_main.xml:

Αποτελεί το xml αρχείο του MainActivity. Σ' αυτό περιγράφεται το layout με βάση το οποίο έχει στηθεί το GUI της εφαρμογής.

Συγκεκριμένα, σ' αυτό το activity περιέχονται τα εξής:

- TextView appName: Ένα απλό TextView που περιέχει το όνομα της εφαρμογής
- TextView welcomeMessage: Ένα απλό TextView που περιέχει ένα μήνυμα προς τον χρήστη.
- Button start: Ένα κουμπί που σε κατευθύνει προς το activity "calculate".
- Button calendar: Ένα κουμπί που σε κατευθύνει προς το activity "calendar".
- Button historyBtn: Ένα κουμπί που σε κατευθύνει προς το activity .

Τα συγκεκριμένα στοιχεία είναι μεταξύ τους συνδεδεμένα μέσω του constraint layout προκειμένου να εξασφαλιστεί η σωστή εμφάνισή τους τόσο σε οριζόντιο όσο και σε κάθετο προσανατολισμό οθόνης.

Κλάση Caldendar.java:

Η κλάση αυτή εμπεριέχει την λειτουργία του Calendar View και των διεργασιών που εκτελούνται μέσα σε αυτό. Χρησιμοποιείται για την προβολή ενός ημερολογίου, με τις ενδείξεις των μετρήσεων του κάθε χρήστη, στην ανάλογη μέρα που εκτελέστηκαν. Η κλάση κληρονομεί χαρακτηριστικά της κλάσης AppCompatActivity. Η λειτουργία αυτής της κλάσης ξεκινάει μόλις ο χρήστης πατήσει το κουμπί Calendar, το οποίο βρίσκεται στο MainActivity. Περιέχει τις εξής μεθόδους:

- protected void onCreate(Bundle savedInstanceState): μέθοδος, η οποία χρησιμοποιείται κατά την εκτέλεση του Activity και έχει τον ρόλο έναρξης της δραστηριότητας. Αποτελεί override μεθόδου της κλάσης-γονέα και χρησιμοποιεί τον κατασκευαστή της. Στην μέθοδο αυτή εκτελείται επίσης override του setOnDateChangeListener του CalendarView, έτσι ώστε σε γεγονός αλλαγής της ημερομηνίας, να αλλάζει την ένδειξη date στο αρχείο activity_calendar.xml στην εκάστοτε ημερομηνία που επιλέγετε και να προβάλλει το ανάλογο γεγονός εκείνης της μέρας

- `public ArrayList<User> getData():` Μία βοηθητική μέθοδος που χρησιμοποιείται για την κατασκευή του `ArrayList` των δεδομένων της βάσης. Αποκτά πρόσβαση στην βάση μέσω ενός αντικειμένου `UserDbManageClass` και κάνει αναζήτηση των εγγραφών μέσω `id` επαναληπτικά (η επανάληψη πραγματοποιείται για αριθμό ίσο με τον αριθμό των εγγραφών). Ελέγχει όλες τις εγγραφές της βάσης και μαρκάρει τις αντίστοιχες ημερομηνίες στο ημερολόγιο. Σε περίπτωση που η ημερομηνία εγγραφής ενός γεγονότος στην βάση, είναι ίδια με αυτή που έχει επιλεγθεί, το γεγονός αυτό προστίθεται στην λίστα ώστε να προβληθεί.

Αρχείο `activity_calendar.xml`:

Αποτελεί το `xml` αρχείο του `Calendar`. Σ' αυτό περιγράφεται το `layout` με βάση το οποίο έχει στηθεί το `GUI` της κλάσης αυτής. Συγκεκριμένα, σ' αυτό το `activity` περιέχονται τα εξής μέσα σε ένα `LinearLayout`:

- `TextView date`: δείχνει την εκάστοτε ημερομηνία
- `LinearLayout` στο οποίο εμπεριέχεται το `MCalendarView`
- `MCalendarView`: κύριο στοιχείο προβολής του ημερολογίου
- `TextView event`: ένδειξη γεγονότος
- `ConstraintLayout` στο οποίο εμπεριέχεται το `RecyclerView`
- `RecyclerView`: κύριο στοιχείο προβολής των γεγονότων

Κλάση `Calculate.java`

Η συγκεκριμένη κλάση αποτελεί ένα απ' τα βασικότερα `activity` της εφαρμογής. Συγκεκριμένα, σ' αυτήν πραγματοποιείται η διεπαφή με τον χρήστη κατά την οποία αυτός/-ή εισάγει τα στοιχεία του και επιτελούνται διάφορες διεργασίες που αναλύονται παρακάτω. Περιέχει τις εξής μεθόδους:

- `public void onCreate(Bundle savedInstanceState):` Μέθοδος που χρησιμοποιείται κατά την έναρξη της δραστηριότητας και αρχικοποιεί τα διάφορα στοιχεία του `GUI`.

- `public void onSaveInstanceState(Bundle outState)`: Μέθοδος που διασφαλίζει την αποθήκευση της κατάστασης του `activity` σε περίπτωση που χρειαστεί να καταστραφεί και να δημιουργηθεί εκ νέου.
- `public void CalculateBmi(View view)`: Αποτελεί την `onClick` μέθοδο του κουμπιού “CALCULATE”. Στην συγκεκριμένη μέθοδο περιέχονται όλες οι παράμετροι υπολογισμού του δείκτη μάζας σώματος. Συγκεκριμένα, περιέχει τόσο ελέγχους προκειμένου να υπάρχει επιβεβαίωση της ορθότητας των δεδομένων που δίνονται (π.χ. έλεγχος ακραίων τιμών ύψους , ύψος = 400 εκατοστά που είναι λάθος τιμή) , αλλά και διάφορα σενάρια που εκτυλίσσονται με βάση τα δεδομένα του χρήστη (π.χ. διαφορετικά αποτελέσματα προκύπτουν με βάση το βάρος , το ύψος αλλά και το φύλο). Τα σενάρια θα γίνουν πιο κατανοητά στο χωρίο των παραδειγμάτων χρήσης της εφαρμογής.
- `public void saveBtn(View view)`: Αποτελεί την `onClick` μέθοδο του κουμπιού “SAVE ENTRY”. Η συγκεκριμένη μέθοδος αναλαμβάνει την διαδικασία αποθήκευσης των δεδομένων της εκτέλεσης του υπολογισμού στην βάση δεδομένων μόνιμης αποθήκευσης. Προκειμένου να υλοποιηθεί η διαδικασία αυτή χρειάζονται στοιχεία που ανήκουν στις κλάσεις της βάσης που αναλύθηκαν παραπάνω. Συγκεκριμένα χρειαζόμαστε ένα αντικείμενο της κλάσης `UserDbManageClass` που υλοποιεί τα SQL ερωτήματα και υποστηρίζει τις διαδικασίες της βάσης. Αποθηκεύουμε τα απαραίτητα δεδομένα που έχουν δοθεί και στην συνέχεια “καθαρίζουμε” το `layout` απ τα παλιά δεδομένα για χρήση απ’ τον επόμενο χρήστη.
- μέθοδος `private void hideWarning(Textview Warning)`: Μία βοηθητική μέθοδος που δέχεται ένα `TextView` και μετά από ένα μικρό χρονικό διάστημα θέτει την ορατότητά του σε “INVISIBLE”. Η χρησιμότητά της αναδεικνύεται παρακάτω.

Αρχείο activity_calculate.xml:

Το συγκεκριμένο αρχείο αποτελεί το .xml αρχείο του activity “Calculate”. Παρακάτω αναλύονται τα περιεχόμενά του. Αναλυτικά αυτό έχει:

- EditText userName: Ένα πεδίο εισαγωγής κειμένου που ο χρήστης δίνει το όνομά του.
- EditText height: Ένα πεδίο εισαγωγής κειμένου που ο χρήστης δίνει το ύψος του.
- EditText weight: Ένα πεδίο εισαγωγής κειμένου που ο χρήστης δίνει το βάρος του.
- ImageView bmiImage: Ένα πεδίο εικόνας που δείχνει εικόνες με βάση τα σενάρια του BMI.
- TextView warning: Ένα πεδίο ανάδειξης κειμένου που βγάζει τα απαραίτητα μηνύματα προς τον χρήστη.
- SwitchCompat switch1: Ένα switch που κάνει toggle ο χρήστης για να δείξει το φύλο του.
- TextView textView: Ένα textView συνοδευτικό προς το switch που “κρατάει” το κείμενο “male”.
- TextView textView2: Ένα textView συνοδευτικό προς το switch που “κρατάει” το κείμενο “female”.
- Button Calculate: Ένα κουμπί που εκτελεί τις διεργασίες υπολογισμού.
- Button saveBtn: Ένα κουμπί που αποθηκεύει τις εκτελέσεις που έχουν προηγηθεί (εφόσον το επιθυμεί ο χρήστης).

Κλάση RecyclerViewAdapter.java:

Η συγκεκριμένη κλάση έχει υλοποιηθεί προκειμένου να υποστηρίξει τις λειτουργίες History και Calendar κατά τις οποίες εμφανίζονται τα δεδομένα της βάσης στον χρήστη. Συγκεκριμένα έχει υλοποιηθεί κάνοντας extend την κλάση “RecyclerView.Adapter<RecyclerView.ViewHolder>”. Πιο συγκεκριμένα αποτελεί Adapter κλάση για ένα card_layout που έχει υλοποιηθεί και σε κάθε car_layout θα τοποθετούνται δεδομένα μίας εγγραφής. Η κλάση αυτή περιέχει:

- `public RecyclerViewAdapter(ArrayList<User> users)`: Ο κατασκευαστής της κλάσης. Δέχεται ένα `ArrayList` που περιέχει αντικείμενα `User`. Κάθε αντικείμενο του `ArrayList` αποτελεί και μία εγγραφή της βάσης δεδομένων.
- Εσωτερική κλάση `ViewHolder` που κάνει `extend` την `“RecyclerView.ViewHolder”` και χρησιμοποιείται, για να ορίσει τα GUI συστατικά που πρέπει να υπάρχουν ανά εγγραφή της βάσης.
- `public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)`: Override μεθόδου της κλάσης - γονέα. Η συγκεκριμένη μέθοδος δημιουργεί ένα `ViewHolder` αντικείμενο και το επιστρέφει.
- `public void onBindViewHolder(NonNull ViewHolder holder, int position)`: Override μεθόδου της κλάσης - γονέα. Η συγκεκριμένη κλάση εξηγεί πως τα δεδομένα του `ArrayList` των χρηστών θα ανατεθούν στα GUI συστατικά του `card_layout`. Το όρισμα `holder` είναι το `ViewHolder` αντικείμενο στο οποίο θα γίνει η ανάθεση και το `int position` είναι ένας ακέραιος που αναδεικνύει την θέση στην οποία βρισκόμαστε (όσον αφορά τον αριθμό των `ViewHolder` που έχουν δημιουργηθεί). Αυτό χρησιμοποιείται για να γίνει ταύτιση του αριθμού ενός `ViewHolder` με την αντίστοιχη εγγραφή της κλάσης.
- `public int getItemCount()`: Override μεθόδου της κλάσης-γονέα. Η μέθοδος αυτή επιστρέφει το πλήθος των εγγραφών που υπάρχουν στο `ArrayList` των `User`.

Κλάση History.java:

Η κλάση αυτή υλοποιεί το activity “History” κατά το οποίο οι εγγραφές της βάσης δεδομένων παρουσιάζονται στον χρήστη με την μορφή καρτών. Η κλάση αυτή περιέχει τα παρακάτω:

- `public void onCreate(Bundle savedInstanceState)`: μέθοδος, η οποία καλείται κατά την ενεργοποίηση του activity. Σ’ αυτήν αρχικοποιούνται τα GUI συστατικά της κλάσης και καλούνται τα απαραίτητα στοιχεία των κλάσεων `RecyclerView`, `RecyclerView.LayoutManager`, `RecyclerView.Adapter`.
- `public ArrayList<User> getData()`: Μία βοηθητική μέθοδος που χρησιμοποιείται για την κατασκευή του `ArrayList` των δεδομένων της βάσης. Αποκτά πρόσβαση στην βάση μέσω ενός αντικειμένου

UserDbManageClass και κάνει αναζήτηση των εγγγραφών μέσω id επαναληπτικά (η επανάληψη πραγματοποιείται για αριθμό ίσο με τον αριθμό των εγγγραφών).

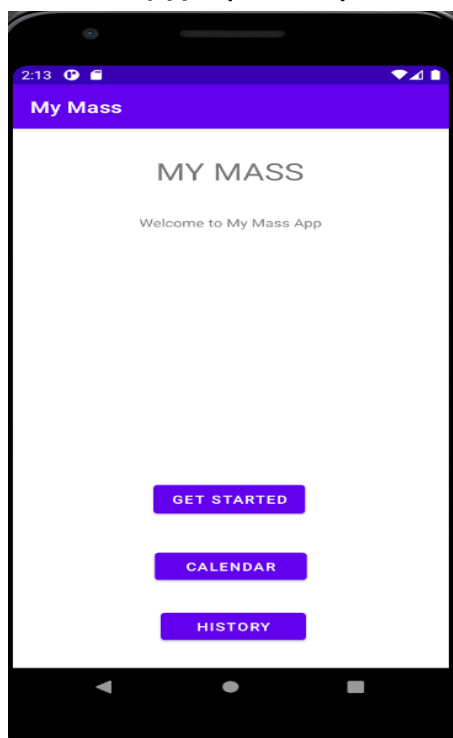
Αρχείο card_layout.xml :

Αυτό το αρχείο αποτελεί ένα .xml αρχείο που αναπαριστά τον τρόπο με τον οποίο δομείται η κάρτα που θα μπαίνουν οι πληροφορίες μιας εγγγραφής. Συγκεκριμένα αυτό περιέχει τα εξής:

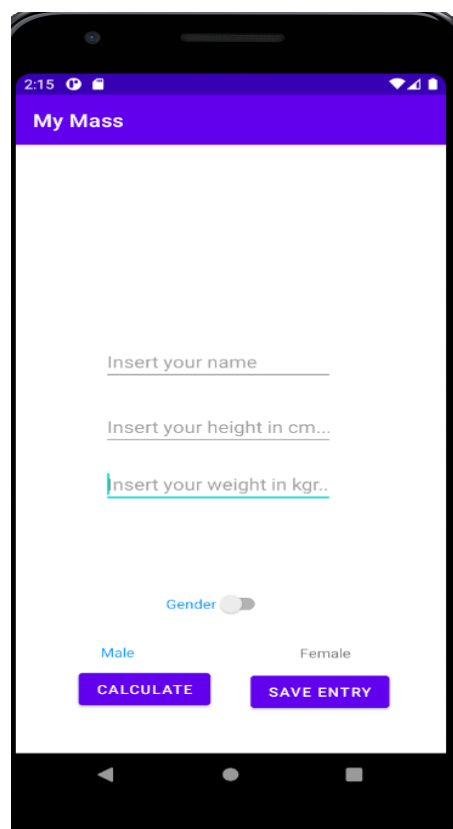
- ImageView bmiImage2: Ένα ImageView που θα παίρνει την εικόνα που πρέπει με βάση το αποθηκευμένο bmi της εγγγραφής.
- TextView userName2: Ένα TextView στο οποίο θα τοποθετείται το userName της εγγγραφής.
- TextView bmiText: Ένα TextView στο οποίο θα τοποθετείται το bmi της εγγγραφής.
- TextView idText: Ένα TextView στο οποίο θα τοποθετείται το id της εγγγραφής.

Screenshot παραδειγμάτων χρήσης:

Αρχική οθόνη



Activity Calculate



Warnings λάθος εισόδων:

The image displays three screenshots of a mobile application titled "My Mass". Each screenshot shows a form with the following fields: a name field (containing "petros"), a height field, a weight field, a gender toggle switch, and two buttons labeled "CALCULATE" and "SAVE ENTRY".

Top Left Screenshot (2:16): The height field contains "500" and the weight field contains "90". A red warning message below the fields reads: "Invalid height value! Change it and try again." The gender toggle is set to "Male".


Top Right Screenshot (2:17): The height field contains "180" and the weight field contains "900". A red warning message below the fields reads: "Invalid weight value! Change it and try again." The gender toggle is set to "Male".

Bottom Screenshot (2:17): The height field contains "1800" and the weight field contains "900". A red warning message below the fields reads: "Invalid weight and height values! Change them and try again." The gender toggle is set to "Male".

Παραδείγματα υπολογισμών

2:21

My Mass



katerina

168

40

BMI: 14.17. You are underweight.
Suggested weight: 58

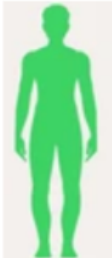
Gender ☒

Male Female

CALCULATE SAVE ENTRY

2:21

My Mass



maria

170

55

BMI: 19.03. You are normal.


Gender ☒

Male Female

CALCULATE SAVE ENTRY

2:19

My Mass



petros

180

150

BMI: 46.30. You are extremely overweight.
Suggested weight: 80


Gender ☐

Male Female

CALCULATE SAVE ENTRY

2:15

My Mass



konstantinos

180

90

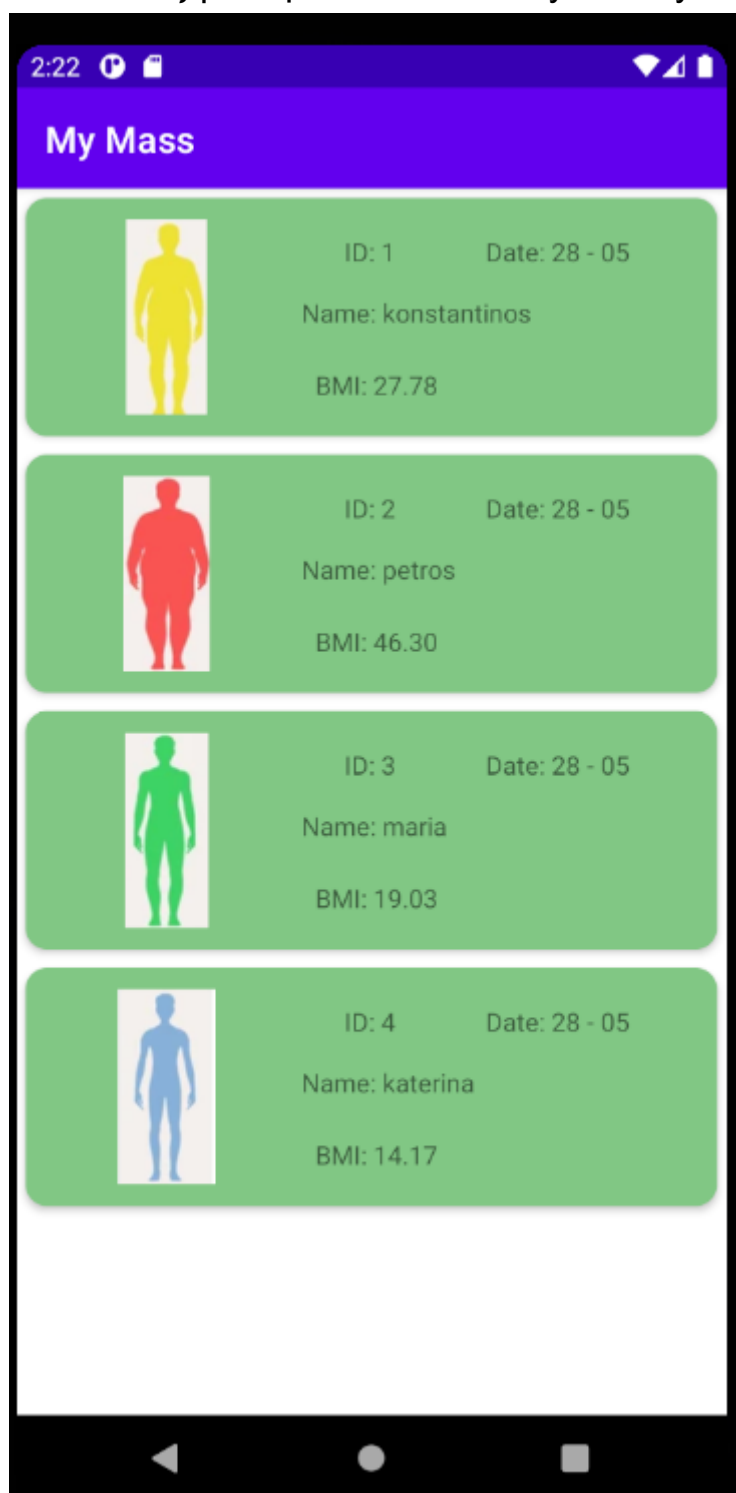
BMI: 27.78. You are overweight.
Suggested weight: 80

Gender ☐

Male Female

CALCULATE SAVE ENTRY

Ανάδειξη ιστορικού στο activity History



Ανάδειξη ιστορικού εντός ημερολογίου στο activity Calendar

