ספר פרויקט DietMaker

בית ספר: מקיף י"א ראשונים

מגיש: אברהם משומר

ת.ז: 328333695

מנחה: טלי רבינוביץ'

תאריך ההגשה: 14.6



תוכן עניינים

[מבוא 4](#_Toc137302527)

[מבנה 5](#_Toc137302528)

[תכנון ותיעוד מסכי הפרויקט 5](#_Toc137302529)

[מסך כניסה 5](#_Toc137302530)

[מסך ABOUT 6](#_Toc137302531)

[מסך HOW TO USE 6](#_Toc137302532)

[מסך הזנת פרטים גופניים 7](#_Toc137302533)

[מסך הזנת פרטי בריאות להיום 9](#_Toc137302534)

[תרשים עמודים 10](#_Toc137302535)

[בחירת מחלקות הפרויקט והקשרים ביניהם – תרשים UML 11](#_Toc137302536)

[מימוש הפרויקט 12](#_Toc137302537)

[StartScreen 12](#_Toc137302538)

[onCreate 12](#_Toc137302539)

[readFromFile 12](#_Toc137302540)

[writeToFile 12](#_Toc137302541)

[AboutScreen 12](#_Toc137302542)

[onCreate 12](#_Toc137302543)

[HowToUseScreen 12](#_Toc137302544)

[onCreate 12](#_Toc137302545)

[DietFirstScreen 12](#_Toc137302546)

[writeToFile 13](#_Toc137302547)

[onCreate 13](#_Toc137302548)

[DietSecondScreen 15](#_Toc137302549)

[assesAllQuality 15](#_Toc137302550)

[assesDiffQuality 15](#_Toc137302551)

[onCreate 15](#_Toc137302552)

[readFromFile 16](#_Toc137302553)

[Quals 17](#_Toc137302554)

[מאגר נתונים 17](#_Toc137302555)

[מדריך למשתמש 18](#_Toc137302556)

[רפלקציה 18](#_Toc137302557)

[נספחים 19](#_Toc137302558)

# מבוא

הפרויקט שלי, DietMaker היא אפצילקציה לאנדרואיד שיוצרת דיאטות אישיות פשוטות המבוססות על נתונים פיזיים. קהל היעד שלי הוא כל אדם שמעוניין להרכיב לעצמו תפריט קלורי וזקוק לסיוע. המחקר שלי כלל קריאה של נושאי בריאות כמו ערך קלורי ממומלץ ליום למגדרים ולגילאים שונים.

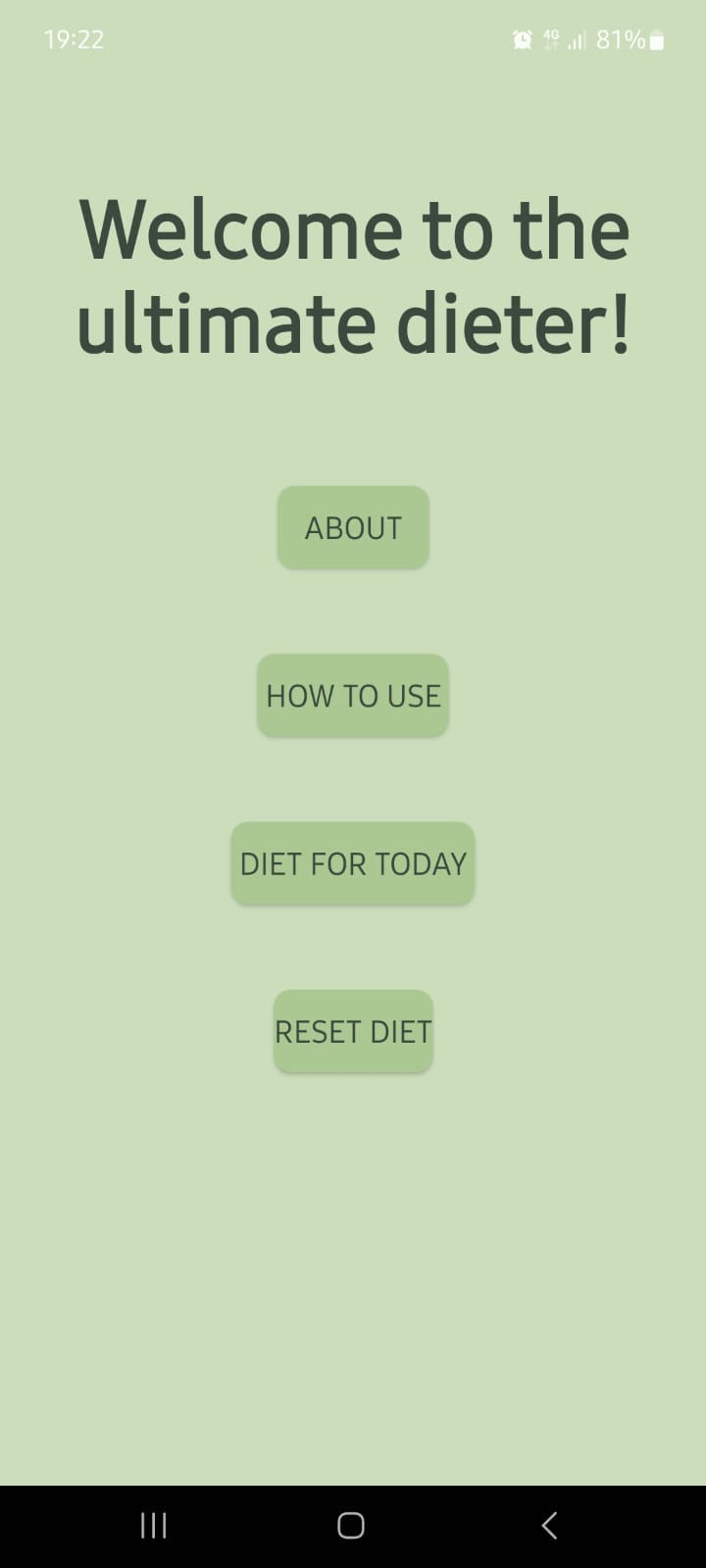
נתקלתי בכל מני בעיות במהלך הפרויקט כמו לחשוב על דרך דירוג ההישג היומי של המשתמש. הייתה לי גם התעכבות על חשיבה על הדרך הפשוטה ביותר שבה משתמש יוכל להזין את הנתונים שלו ולקבל את הדיאטה שלו בדרך הנוחה ביותר. הפתרון שלי היה מספר מסכים שמהווים התקדמות בהזנת הפרטים.

# מבנה

## תכנון ותיעוד מסכי הפרויקט

### מסך כניסה

זה המסך שנפתח לראשונה באפליקציה, הוא מכיל 4 כפתורים:



כפתור שמוחק את פרטי המשתמש

כפתור שמעביר אל מסך עם פרטים על האפליקציה

כפתור שמעביר אל מסך הוראות שימוש

כפתור שמעביר אל מסך הזנת פרטים גופניים/עמוד פרטי בריאות להיום

### מסך ABOUT

בעמוד הזה מופיע טקסט עם פרטים על האפליקציה.

תמונה שמכילה טקסט, צילום מסך, גופן

התיאור נוצר באופן אוטומטי

### מסך HOW TO USE

במסך זה מופיע טקסט עם הוראות שימוש לאפליקציה.

תמונה שמכילה טקסט, צילום מסך, גופן

התיאור נוצר באופן אוטומטי

### מסך הזנת פרטים גופניים

במסך זה מזינים את הפרטים הגופניים של המשתמש.

תמונה שמכילה טקסט, צילום מסך, מחשבון

התיאור נוצר באופן אוטומטי

כפתור שמעביר למסך הזנת פרטי בריאות להיום

בחירה בין זכר ונקבה

שדה הזנת גיל

שדה הזנת משקל

שדה הזנת גובה

### מסך הזנת פרטי בריאות להיום

תמונה שמכילה טקסט, צילום מסך, גופן, מסמך

התיאור נוצר באופן אוטומטי

עזר חזותי כדי להבין את מצב ה-BMI של המשתמש

טקסט הערכה, משתנה בהתאם לנתונים שהמשתמש הכניס במסך

טקסט עם מידע על הנתונים של המשתמש מבחינת BMI

גולל שבערתו מזינים כמות שעות פעילות גופנית

גולל שבעזרתו אפשר להזין כמות קלוריות

## תרשים עמודים

מסך כניסה

מסך ABOUT

מסך HOW TO USE

מסך הזנת פרטים אישיים

מסך הזנת פרטי בריאות להיום

## PlantUML diagramבחירת מחלקות הפרויקט והקשרים ביניהם – תרשים UML

# מימוש הפרויקט

## StartScreen

מממשת את מסך הכניסה.

### onCreate

הפעולה ההתחלתית של המסך. שמה על הכפתורים את הניתוב למסכים הייעודיים להם. על כפתור DIET FOR TODAY שמה את מסך הזנת פרטים אישיים אם אין קובץ שמכיל מידע על המשתמש/יש קובץ בלי מידע, אחרת שמה את מסך הזנת פרטי בריאות להיום – הבדיקה נעשית באמצעות readFromFile. שמה שבלחיצה על כפתור RESET DIET קובץ המידע על המשתמש יאותחל באמצעות קריאה ל- writeToFile.

### readFromFile

פעולה שבאמצעותה נגשים לקובץ (לפי שמו – פרמטר) ומחזירים את תכנו כמחרוזת.

### writeToFile

כותבת לקובץ (לפי שמו – פרמטר) את המידע שנותנים (כפרמטר).

## AboutScreen

מממשת את מסך הפרטים על האפליקציה.

### onCreate

הפעולה ההתחלתית של המסך. מציגה את הטקסט הרלוונטי על האפליקציה באמצעות קובץ המחרוזות.

## HowToUseScreen

מממשת את מסך הוראות השימוש של האפליקציה.

### onCreate

הפעולה ההתחלתית של המסך. מציגה את הטקסט הרלוונטי על שימוש באפליקציה באמצעות קובץ המחרוזות.

## DietFirstScreen

מממשת את מסך הזנת פרטי המשתמש.

### writeToFile

כותבת לקובץ (לפי שמו – פרמטר) את המידע שנותנים (כפרמטר).

### onCreate

הפעולה ההתחלתית של המסך. שמה על כפתור החישוב את הפעולות הבאות:

* חישוב של ה-BMI של המשתמש לפי הנוסחה:
  + BMI = weight / ((height/100)^2)
* חישוב כמות הקלוריות של המשתמש לפי נוסחת האריס-בנדיקט:
  + לגברים: 13.397 \* weight + 4.799 \* height - 5.677 \* age + 88.362
  + לנשים: 9.247 \* weight + 3.098 \* height - 4.330 \* age + 447.593
* חישוב מצב הבריאות של המשתמש וכמות הפעילות הגופנית הרצויה למשתמש:

תת משקל

0.5 שעות אימון

בריא

1 שעות אימון

עודף משקל

2 שעות אימון

השמנת יתר

2 שעות אימון

20

18.5

30

BMI

* כתיבת כל הנתונים האלה לקובץ "DATA.txt" על הטלפון באמצעות writeToFile.
* מעבר לעמוד הזנת פרטי בריאות להיום.



## DietSecondScreen

מממשת את מסך הזנת פרטי הבריאות להיום.

### assesAllQuality

מחזירה מחרוזת לפי הממוצע של cqual ו-wqual:

* 0 -> "רע מאוד"
* 0.5 -> "רע"
* 1 -> "לא טוב"
* 1.5 -> "טוב"
* 2 -> "מצוין"

### assesDiffQuality

מקבלת את ההפרש בין הקלט לבין הכמות הרצויה, ואת הכמות הרצויה, ומחזירה:

* 0 – אם ההפרש בין הקלט לבין הכמות הרצויה עולה על 2/3 מהכמות הרצויה
* 1 – אם ההפרש בין הקלט לבין הכמות הרצויה עולה על 1/4 מהכמות הרצויה
* 2 – אחרת

### onCreate

הפעולה ההתחלתית של המסך. מבצעת את הפעולות הבאות:

* קוראת את הנתונים של המשתמש מקובץ "DATA.txt" באמצעות readFromFile.
* לפי הנתונים שמה את הטקסט של צרכי היום למשתמש על המסך: "אתה צריך לאכול ... קלוריות היום, ולהתאמן ... שעות".
* שמה את נתוני הBMI- לעזר החזותי.
* בהזזת גולל הקלוריות ובהזזת גולל שעות האימון:
  + מחושב היחס בין הקלט לעומת הכמויות הרצויות שחושבו במסך הקודם.
  + לפי היחס מתקבל משוב בתיבות הטקסט מתחת לגוללים: "'קצת מדי/בדיוק/יותר מדי".
  + שמים ב-cqual/wqual של Quals את האיכות של הערך לעומת הקלט מהמשתמש באמצעות assesDiffQuality.
  + אם הממוצע של cqual ו-wqual, שם את הטקסט בטקסט ההערכה בתחתית המסך באמצעות assesAllQuality.



### readFromFile

פעולה שבאמצעותה נגשים לקובץ (לפי שמו – פרמטר) ומחזירים את תכנו כמחרוזת.

## Quals

מחלקה סטטית למשתנים גלובליים:

* cqual – איכות הקלט של הקלוריות לעומת הכמות הרצויה.
* wqual – איכות הקלט של שעות האימון לעומת הכמות הרצויה.

## מאגר נתונים

מאגר הנתונים הוא באמצעות קובץ שנשמר בטלפון בשם "DATA.txt". משתמשים בו כדי לשמור את הנתונים הפיזיים של המשתמש.

# מדריך למשתמש

כדי להריץ את הפרויקט יש לפתוח אותו ב-Andorid Studio ולהריץ באמצעות אימולטור או מכשיר שמחובר למחשב. הפרויקט נבדק על מכשיר SAMSUNG Galaxy A51. הפרויקט רץ על כל מכשיר אנדרואיד עם גרסה 11 ומעלה.

# רפלקציה

בשבילי העבודה על הפרויקט הייתה די פשוטה, חוץ מהחלק שלא כל כך החלטתי מה אני רוצה לעשות עד מאוחר יחסית. כשהתחלנו לעבוד על האפליקציה אני חשבתי במשך זמן רב מה יהיה הנושא ואיך לעשות את הצעת הפרויקט. לאחר שסיימתי את הצעת הפרויקט והמורה אישר לי את הרעיון התחלתי לעשות סקיצה לאפליקציה. לקח לי כמה זמן להבין איך בדיוק אני רוצה שזה ייראה ואיך המסכים יעבדו ביניהם. במיוחד הסתבכתי עם איך לאחסן את הנתונים (בחרתי בסוף בצורת קובץ ולא במאגר נתונים).

למדתי מהפרויקט איך לדחוק בעצמי ולנסות לעשות המון דברים בזמן קצר, איך להתגבר על מכשולים ועל המון חיפושים בגוגל כדי לעשות דברים שלא למדנו בכיתה.

לו הייתי מתחיל לעבוד על הפרויקט היום, הייתי לגמרי חושב יותר על מתי צריך להגיש וכמה זמן ייקח לי להכין כל דבר. גם הייתי יותר חושב על איך המסכים יראו יותר כי היו לי ספקות ושינויים רבים בזמן שכתבתי את הקוד.

# נספחים

public class StartScreen extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle *savedInstanceState*) {

        super.onCreate(*savedInstanceState*);

        setContentView(R.layout.activity\_start\_screen);

        getSupportActionBar().hide(); *// hides the action bar*

        Button aboutButton = (Button) findViewById(R.id.AboutButton); *// button for screen w/ info about project*

        Button howToUseButton = (Button) findViewById(R.id.HowToUseButton); *// button for screen w/ info about  how to use the app*

        Button enterDietButton = (Button) findViewById(R.id.EnterDietButton); *// button for screen for physical data input or output*

        Button resetDietButton = (Button) findViewById(R.id.ResetDietButton); *// button for resetting diet data*

        enterDietButton.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View *view*) {

*// when tapping on enterDietButton*

*// leads to DietFirstScreen activity if there is no diet data, if there is then leads to DietSecondScreen activity.*

                String data = readFromFile("DATA"); *// reading data*

                if (data == "") {startActivity(new Intent(getApplicationContext(), DietFirstScreen.class));}

                else {startActivity(new Intent(getApplicationContext(), DietSecondScreen.class));}

            }

        });

        aboutButton.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View *view*) {

*// leads to AboutScreen activity*

                startActivity(new Intent(getApplicationContext(), AboutScreen.class));

            }

        });

        howToUseButton.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View *view*) {

*// leads to HowToUseScreen activity*

                startActivity(new Intent(getApplicationContext(), HowToUseScreen.class));

            }

        });

        resetDietButton.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View *view*) {

*// resetting data by rewriting the data file*

                writeToFile("DATA", "", "w");

            }

        });

    }

    public String readFromFile(String *filename*) {

*// receive a string containing a file's name.*

*// reads string from file*

        String ret = "";

        try {

*// entering file*

            InputStream inputStream = openFileInput(*filename*+".txt");

            if ( inputStream != null ) {

*// as long as file has a value*

                InputStreamReader inputStreamReader = new InputStreamReader(inputStream);

                BufferedReader bufferedReader = new BufferedReader(inputStreamReader);

                String receiveString = "";

                StringBuilder stringBuilder = new StringBuilder();

                while ( (receiveString = bufferedReader.readLine()) != null ) {

*// as there is still text ahead*

*// appends the next text, byte by byte*

                    stringBuilder.append(receiveString);

                }

*// closing file*

                inputStream.close();

*// bytes to string*

                ret = stringBuilder.toString();

            }

        }

        catch (FileNotFoundException *e*) {

*// send error to be shown in ErrorScreen activity*

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "File not found: " + e.toString());

            startActivity(intent);

        } catch (IOException *e*) {

*// send error to be shown in ErrorScreen activity*

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "Can not read file: " + e.toString());

            startActivity(intent);

        }

        return ret;

    }

    public void writeToFile(String *filename*, String *data*, String *mode*) {

*// recieves string filename, data as string and mode as string*

*// write data to file w/ filename according to mode*

        try {

            FileOutputStream fos = null;

*// "w" => rewrite file w/ data. "a" => appending data to file.*

            if (*mode*.equals("w")) {fos = openFileOutput(*filename*+".txt", MODE\_PRIVATE);}

            if (*mode*.equals("a")) {fos = openFileOutput(*filename*+".txt", MODE\_APPEND);}

*// entering file*

            OutputStreamWriter outputStreamWriter = new OutputStreamWriter(fos);

            outputStreamWriter.write(*data*); *// writing data to file*

            outputStreamWriter.close(); *// closing file*

        }

        catch (Exception *e*) {

*// send error to be shown in ErrorScreen activity*

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "File write failed: " + e.toString());

            startActivity(intent);

        }

    }

}

public class AboutScreen extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle *savedInstanceState*) {

        super.onCreate(*savedInstanceState*);

        setContentView(R.layout.activity\_about\_screen);

        getSupportActionBar().hide(); *// hides the action bar*

*// viewing the required text in a text view.*

        TextView tv = (TextView) findViewById(R.id.AboutTextView);

        tv.setText(getString(R.string.aboutText));

        tv.setTextSize(30);

    }

}

public class HowToUseScreen extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle *savedInstanceState*) {

        super.onCreate(*savedInstanceState*);

        setContentView(R.layout.activity\_how\_to\_use\_screen);

        getSupportActionBar().hide(); *// hides the action bar*

*// viewing the required text in a text view.*

        TextView tv = (TextView) findViewById(R.id.HowToUseTextView);

        tv.setText(getString(R.string.HowToUseText));

        tv.setTextSize(25);

    }

}

public class DietFirstScreen extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle *savedInstanceState*) {

        super.onCreate(*savedInstanceState*);

        setContentView(R.layout.activity\_diet\_first\_screen);

        getSupportActionBar().hide(); *// hides the action bar*

        Button calcDietButton = (Button) findViewById(R.id.CalcDietButton);

*// all the actions takes place when the user's tapping calcDietButton*

        calcDietButton.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View *view*) {

*// the gender spinnre*

                Spinner genderSpinner = (Spinner) findViewById(R.id.GenderSpinner);

*// getting the gender as string from it.*

                final String gender = genderSpinner.getSelectedItem().toString();

                EditText heightEditText = (EditText) findViewById(R.id.HeightEditText);

                EditText weightEditText = (EditText) findViewById(R.id.WeightEditText);

                EditText ageEditText = (EditText) findViewById(R.id.AgeEditText);

*// getting the physical data from EditTexts*

                double height = Double.parseDouble(heightEditText.getText().toString());

                double weight = Double.parseDouble(weightEditText.getText().toString());

                double age = Double.parseDouble(ageEditText.getText().toString());

*// setting bmi by the BMI formula*

                double bmi = weight / Math.pow(height / 100, 2);

                bmi = Double.parseDouble(String.format("%.2f", bmi)); *// limiting it to 2 digits after dot.*

*// setting calories*

                int calories = 0;

*// formula from https://www.calculator.net/calorie-calculator.html*

                if (gender.equals("Male")) {

                    calories = (int) (13.397 \* weight + 4.799 \* height - 5.677 \* age + 88.362);

                }

                if (gender.equals("Female")) {

                    calories = (int) (9.247 \* weight + 3.098 \* height - 4.330 \* age + 447.593);

                }

*// setting the state and workout time by the bmi value*

                String state = " ";

                double wrkout = 0.0;

                if (bmi < 18.5) {

                    state = "UNDERWEIGHT";

                    wrkout = 0.5;

                }

                if (bmi >= 18.5 && bmi <= 20) {

                    state = "Healthy";

                    wrkout = 1;

                }

                if (bmi > 20 && bmi < 30) {

                    state = "OVERWEIGHT";

                    wrkout = 2;

                }

                if (bmi >= 30) {

                    state = "OBESE!";

                    wrkout = 3;

                }

*// writing to file all data to be used by DietSecondScreen activity*

                writeToFile("DATA", bmi + "", "w");

                writeToFile("DATA", "-" + height, "a");

                writeToFile("DATA", "-" + weight, "a");

                writeToFile("DATA", "-" + age, "a");

                writeToFile("DATA", "-" + gender, "a");

                writeToFile("DATA", "-" + state, "a");

                writeToFile("DATA", "-" + calories, "a");

                writeToFile("DATA", "-" + wrkout, "a");

*// then goes to it*

                startActivity(new Intent(getApplicationContext(), DietSecondScreen.class));

            }

        });

    }

    public void writeToFile(String *filename*, String *data*, String *mode*) {

*// recieves string filename, data as string and mode as string*

*// write data to file w/ filename according to mode*

        try {

            FileOutputStream fos = null;

*// "w" => rewrite file w/ data. "a" => appending data to file.*

            if (*mode*.equals("w")) {fos = openFileOutput(*filename*+".txt", MODE\_PRIVATE);}

            if (*mode*.equals("a")) {fos = openFileOutput(*filename*+".txt", MODE\_APPEND);}

*// entering file*

            OutputStreamWriter outputStreamWriter = new OutputStreamWriter(fos);

            outputStreamWriter.write(*data*); *// writing data to file*

            outputStreamWriter.close(); *// closing file*

        }

        catch (Exception *e*) {

*// send error to be shown in ErrorScreen activity*

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "File write failed: " + e.toString());

            startActivity(intent);

        }

    }

}

public class DietSecondScreen extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle *savedInstanceState*) {

        super.onCreate(*savedInstanceState*);

        setContentView(R.layout.activity\_diet\_second\_screen);

        getSupportActionBar().hide(); *// hides the action bar*

*// getting the data from the data file*

        String data = readFromFile("DATA");

        String[] vars = data.split("-"); *// splitting it to a string array*

        double bmi = Double.parseDouble(vars[0]);

        double height = Double.parseDouble(vars[1]);

        double weight = Double.parseDouble(vars[2]);

        double age = Double.parseDouble(vars[3]);

        String gender = vars[4];

        String state = vars[5];

        final int calories = Integer.parseInt(vars[6]);

        final double workout = Double.parseDouble(vars[7]);

*// setting the title*

        String diet2Title = String.format(getResources().getString(R.string.Diet2Title), state, calories, workout);

        TextView titv = (TextView) findViewById(R.id.TitleInfoTextView);

        titv.setText(diet2Title);

*// setting title seek bar which is only for show.*

        SeekBar stateSeekBar = (SeekBar) findViewById(R.id.StateSeekBar);

        stateSeekBar.setMax(40 \* 100);

        stateSeekBar.setProgress((int) (bmi \* 100));

        stateSeekBar.setEnabled(false);

*// setting the calories seek bar w/ values 0~calories\*2*

*// and the workout time seek bar w/ values 0~40*

        SeekBar caloriesSeekBar = (SeekBar) findViewById(R.id.CaloriesSeekBar);

        SeekBar workoutSeekBar = (SeekBar) findViewById(R.id.WorkoutSeekBar);

        caloriesSeekBar.setMax(calories \* 2);

        workoutSeekBar.setMax(40);

*// when the calories seek bar is changed*

        caloriesSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {

            @Override

            public void onProgressChanged(SeekBar *seekBar*, int *progress*, boolean *fromUser*) {

*// views the current value of seek bar below it.*

                TextView caloriesTextView = (TextView) findViewById(R.id.CaloriesTextView);

                if(*progress* == *seekBar*.getMax())

                    caloriesTextView.setText(*progress* + "+");

                else

                    caloriesTextView.setText("" + *progress*);

*// location:*

*//int val = (progress \* (seekBar.getWidth() - 2 \* seekBar.getThumbOffset())) / seekBar.getMax();*

*//textView.setX(seekBar.getX() + val + seekBar.getThumbOffset() / 2);*

*// tells the user if he ate too little~too much by the seek bar progress.*

                TextView caloriesResponseTextView = (TextView) findViewById(R.id.CaloriesResponseTextView);

                if (*progress* < 0.75\*calories)

*// too little*

                    caloriesResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[0]);

                else if (*progress* > 1.25\*calories)

*// too much*

                    caloriesResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[2]);

                else

*// fine*

                    caloriesResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[1]);

*// calculating the difference between progress and the required calories*

                double caloriesDiff = Math.abs(calories-*progress*);

                Quals.cqual = assesDiffQuality(caloriesDiff, calories); *// put it in Quals.cqual*

*// show in summary text view of screen the overall assessments of today's work.*

                TextView summaryMessageTextView = findViewById(R.id.Diet2SummaryMessageTextView);

                String summaryMessage = getResources().getString(R.string.Diet2SummaryMessage);

                summaryMessage = String.format(summaryMessage, assesAllQuality());

                summaryMessageTextView.setText(summaryMessage);

            }

            @Override

            public void onStartTrackingTouch(SeekBar *seekBar*) {

            }

            @Override

            public void onStopTrackingTouch(SeekBar *seekBar*) {

            }

        });

*// when the workout seek bar is changed*

        workoutSeekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {

            @Override

            public void onProgressChanged(SeekBar *seekBar*, int *progress*, boolean *fromUser*) {

*// views the current value of seek bar below it.*

*// the workout time is required to be a double, so it will be divide by 10: 0~40 => 0~4 hours.*

                TextView workoutTextView = (TextView) findViewById(R.id.WorkoutTextView);

                if(*progress* == *seekBar*.getMax())

                    workoutTextView.setText(((double)(*progress*)/10) + "+");

                else

                    workoutTextView.setText("" + (double)(*progress*)/10);

*// location:*

*//int val = (progress \* (seekBar.getWidth() - 2 \* seekBar.getThumbOffset())) / seekBar.getMax();*

*//textView.setX(seekBar.getX() + val + seekBar.getThumbOffset() / 2);*

*// tells the user if he ate too little~too much by the seek bar progress.*

                TextView workaoutResponseTextView = (TextView) findViewById(R.id.WorkoutResponseTextView);

                if ((double)(*progress*)/10 < 0.75\*workout)

*// too little*

                    workaoutResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[0]);

                else if ((double)(*progress*)/10 > 1.25\*workout)

*// too much*

                    workaoutResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[2]);

                else

*// fine*

                    workaoutResponseTextView.setText(getResources().getStringArray(R.array.ResponseMessage)[1]);

*// calculating the difference between progress and the required workout hours*

                double workoutDiff = Math.abs(workout-(double)(*progress*)/10);

                Quals.wqual = assesDiffQuality(workoutDiff, workout);*// put it in Quals.cqual*

*// show in summary text view of screen the overall assessments of today's work.*

                TextView summaryMessageTextView = findViewById(R.id.Diet2SummaryMessageTextView);

                String summaryMessage = getResources().getString(R.string.Diet2SummaryMessage);

                summaryMessage = String.format(summaryMessage, assesAllQuality());

                summaryMessageTextView.setText(summaryMessage);

            }

            @Override

            public void onStartTrackingTouch(SeekBar *seekBar*) {

            }

            @Override

            public void onStopTrackingTouch(SeekBar *seekBar*) {

            }

        });

    }

    public String readFromFile(String *filename*) {

        String ret = "";

        try {

            InputStream inputStream = openFileInput(*filename*+".txt");

            if ( inputStream != null ) {

                InputStreamReader inputStreamReader = new InputStreamReader(inputStream);

                BufferedReader bufferedReader = new BufferedReader(inputStreamReader);

                String receiveString = "";

                StringBuilder stringBuilder = new StringBuilder();

                while ( (receiveString = bufferedReader.readLine()) != null ) {

                    stringBuilder.append(receiveString);

                }

                inputStream.close();

                ret = stringBuilder.toString();

            }

        }

        catch (FileNotFoundException *e*) {

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "File not found: " + e.toString());

            startActivity(intent);

        } catch (IOException *e*) {

            Intent intent = new Intent(getBaseContext(), ErrorScreen.class);

            intent.putExtra("ErrorMessage2", "Can not read file: " + e.toString());

            startActivity(intent);

        }

        return ret;

    }

    public int assesDiffQuality(double *diff*, double *goodVal*){

*// receives the difference between a value and goodVal, and goodVal*

*// assessing the quality of the difference relatively to goodVal*

*// return the quality as int 0~2*

        if(*diff* > 0.625\**goodVal*)

*// very bad*

            return 0;

        else if(*diff* > 0.25\**goodVal*)

*// not so good*

            return 1;

        else

*// great*

            return 2;

    }

    public String assesAllQuality(){

*// returns the average quality of Quals' vars as string words.*

        double final\_qual = (Quals.cqual+Quals.wqual)/2; *// average*

        if (final\_qual == 0.0)

            return "very bad";

        if (final\_qual == 0.5)

            return "bad";

        if (final\_qual == 1.0)

            return "not so good";

        if (final\_qual == 1.5)

            return "good";

        if (final\_qual == 2.0)

            return "great";

        return "";

    }

}

class Quals {

*// a class for the global integers cqual and wqual, representing calories' consumption quality*

*// and workout time quality, respectively.*

    public static int cqual = 0;

    public static int wqual = 0;

}