Project Title:

Group members: Konrad Gnoiński (kogno14@student.sdu.dk), David Bagdasaryan (dbagd14@student.sdu.dk), Milena Gnoińska (migno14@student.sdu.dk).

Introduction:

Dak-E is an iOS app for medical clinics. User is supposed to swipe yellow card and then fill in a questionare about his/her lifestyle. Data filled by user will be updated to a central database. Basic idea is to make it as fast and simple as possible, for a wide range of variety users to fill a questionare. This subject is very important because it will help to detect illnesses in a faster way, with less attention from a doctor. Those illnesses could be detected faster because that way we will get date more often, witch is most crucial part - from symptoms to detect an illness. Main problem is that we wish to use a time of a patient, that time that normally is being wasted in waiting room, to get some more data about patient. We are working with DAK-E. The objective of DAK-E Danish Quality Unit of General Practice – is to improve the quality work of general practice. DAK-E has asked us for help because that wanted to get a prototype of their idea. For more informations about company you can go to: http://www.dak-e.dk.
Our objectives are pretty simple - we have to make a questionnaire based on raw data from database - it could be about anything. This questionnaire should be easy to fill and don't need any maintenance on code to make new questionaries. In later part the answers from user should be presented to make them sure that the answer are correct.

Functional and non-functional requirements

All of requirements listed beneath were given by DAK-E

- application should be easy to use by anyone, not matter what age a person is
- there should be a administration part with capability of changing colours of UI items and switching between different questionaries
- application should look well on iPad mini
- user should be able to login in by swiping a yellow card
- when logging in data from card should be verified

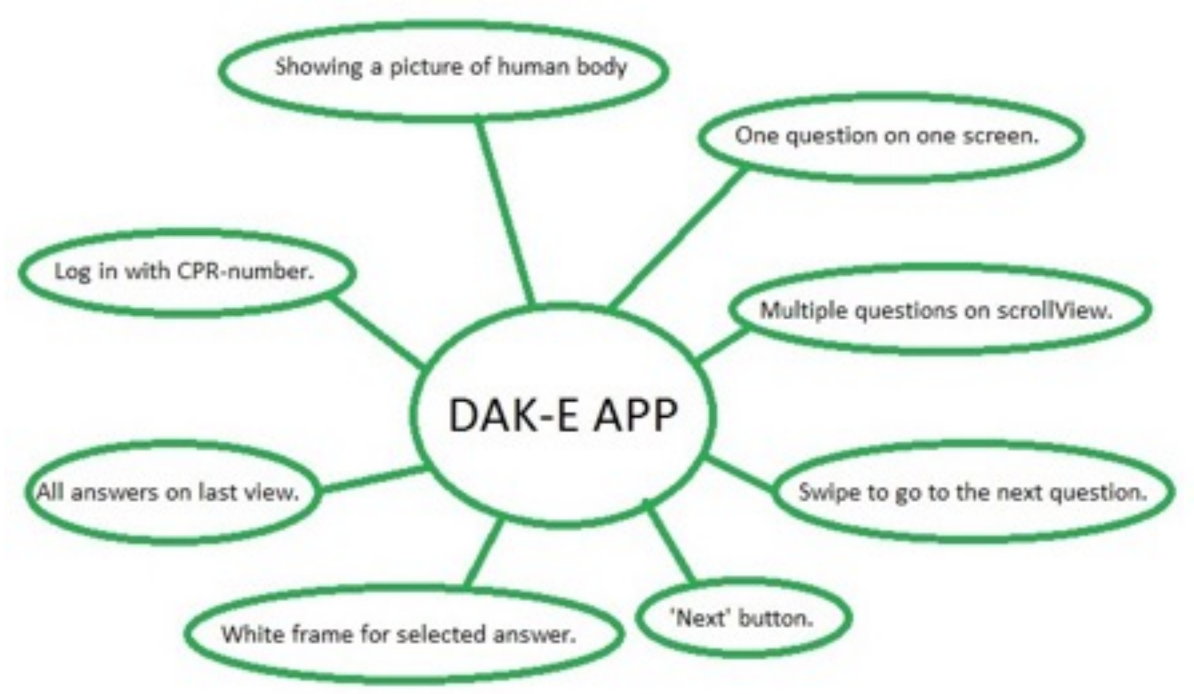
- application should create dynamically interactive questionnaire based on a data downloaded via network from a remote database
- just before end, all answers should be displayed and wait for confirmation by the user
- after filling a questionnaire application should send a data to remote database
- in case of no connection data should be stored in local database waiting for synchronisation

Methods

- Brainstorm
- Simple Prototyping
- Evaluation
- Building a first navigational digital prototype
- Evaluation
- Use-case diagrams
- Object diagrams
- Finding requirements
- Implementation
    - Frameworks used
    - Classes (model- view controller)
    - Properties and Methods
- Final evaluation
- Bugs fixing

Results

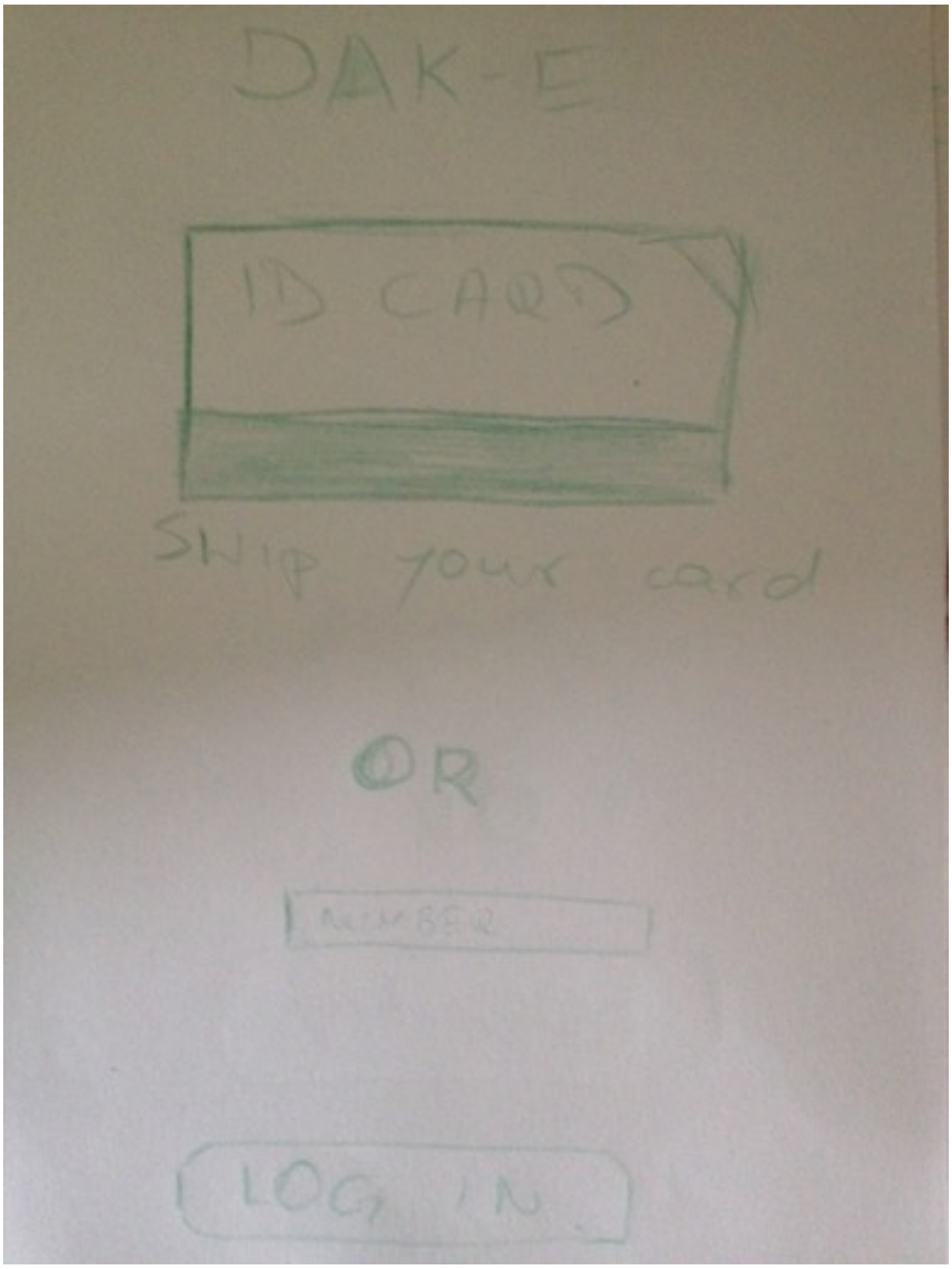
- Brainstorm
  We started with brainstorming to discuss all the ideas we had. We created a mind map.


Showing a picture of human body
One question on one screen.
Log in with CPR-number.
Multiple questions on scrollView.
DAK-E APP
All answers on last view.
Swipe to go to the next question.
'Next' button.
White frame for selected answer.

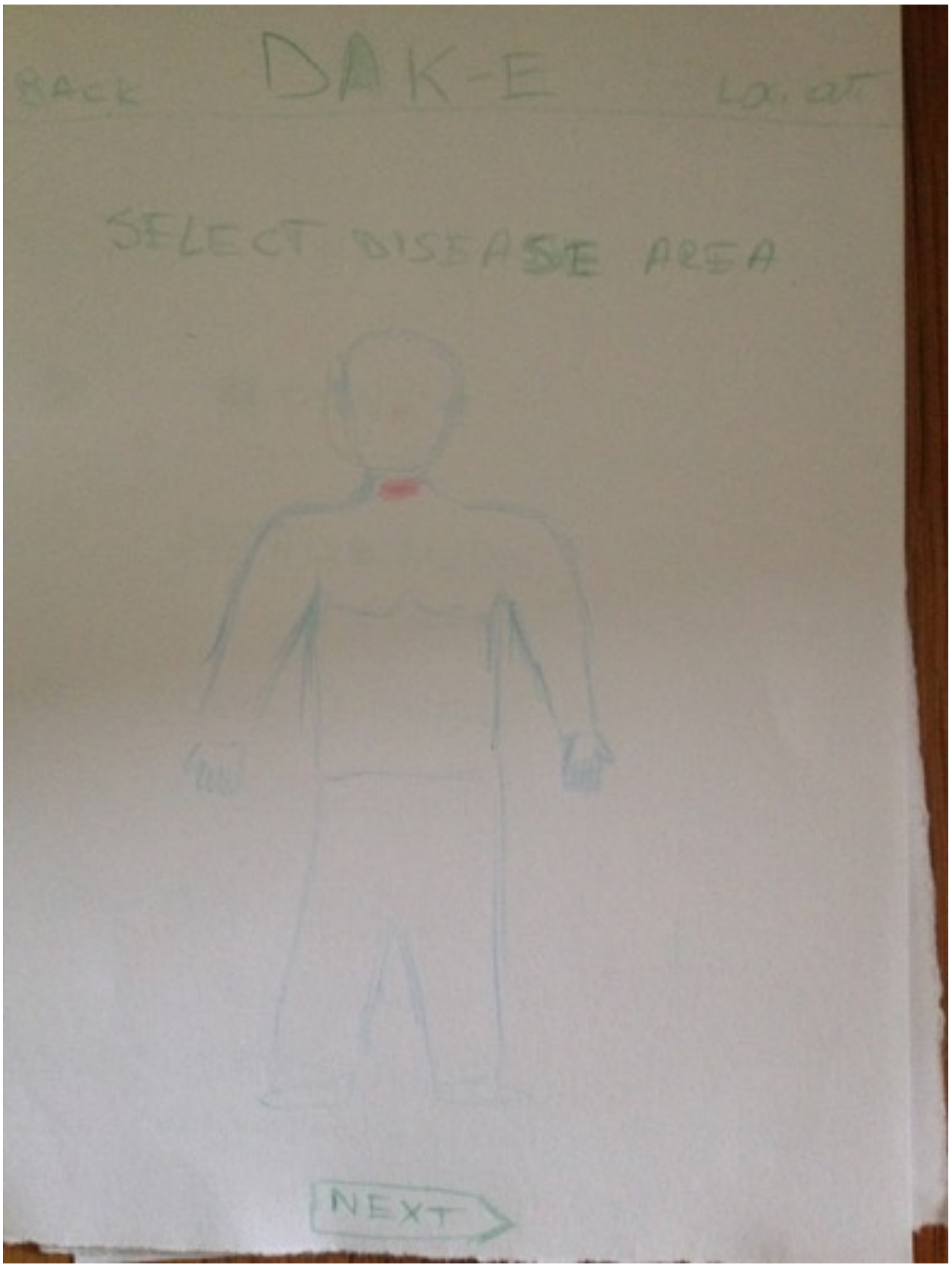
We rejected the idea of swiping to go to the next question because we thought older people may have problems with that. It may also cause that people would skip some questions.
We decided not to put multiple questions on scrollView because it seemed to us not to be so intuitive and could also cause skipping questions.

- Simple prototyping
  We started design by preparing paper prototype. We created five draws with possible views.

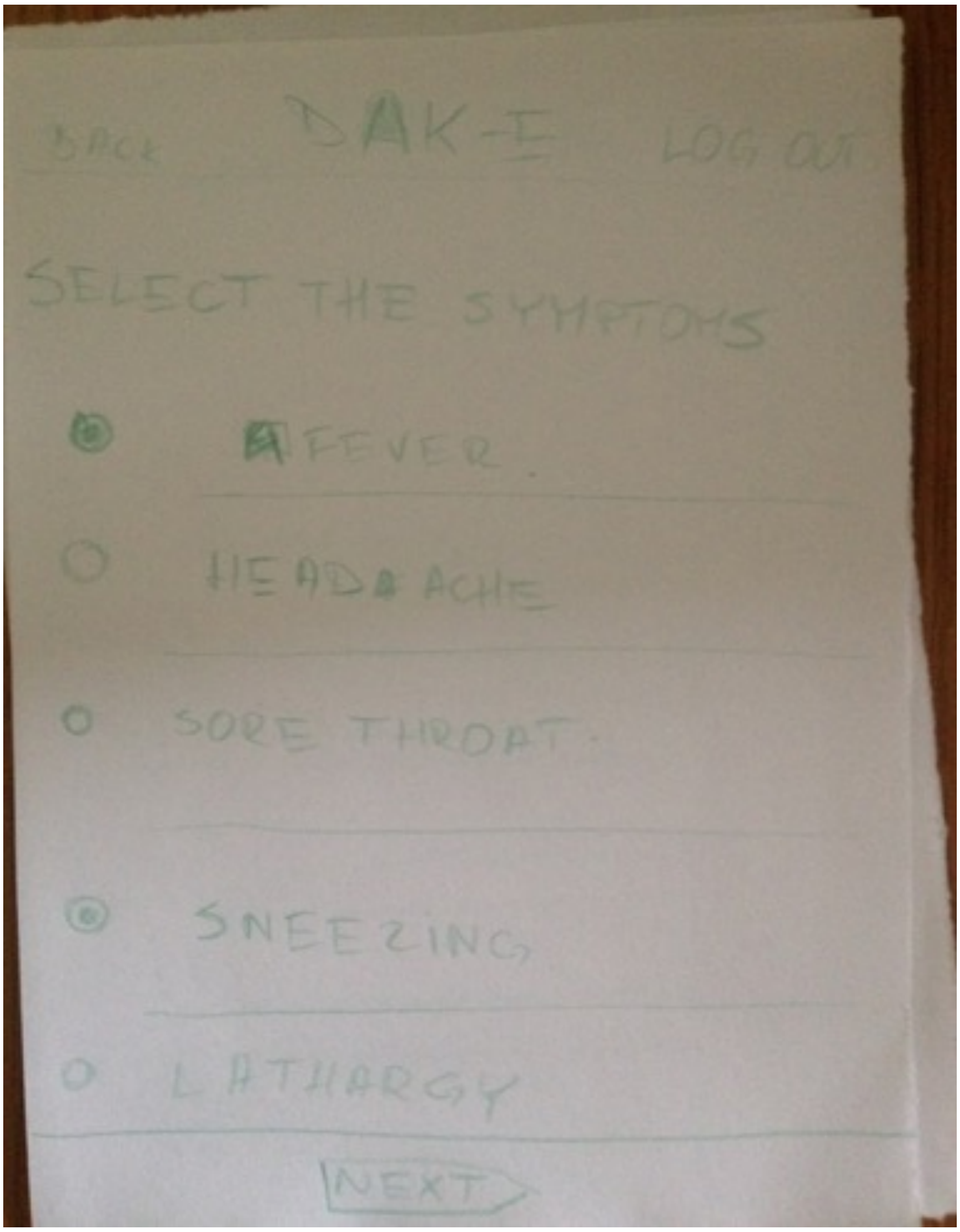
  This is the first screen. It tells user how to log in. User can log in by swiping yellow card or typing cpr-number.



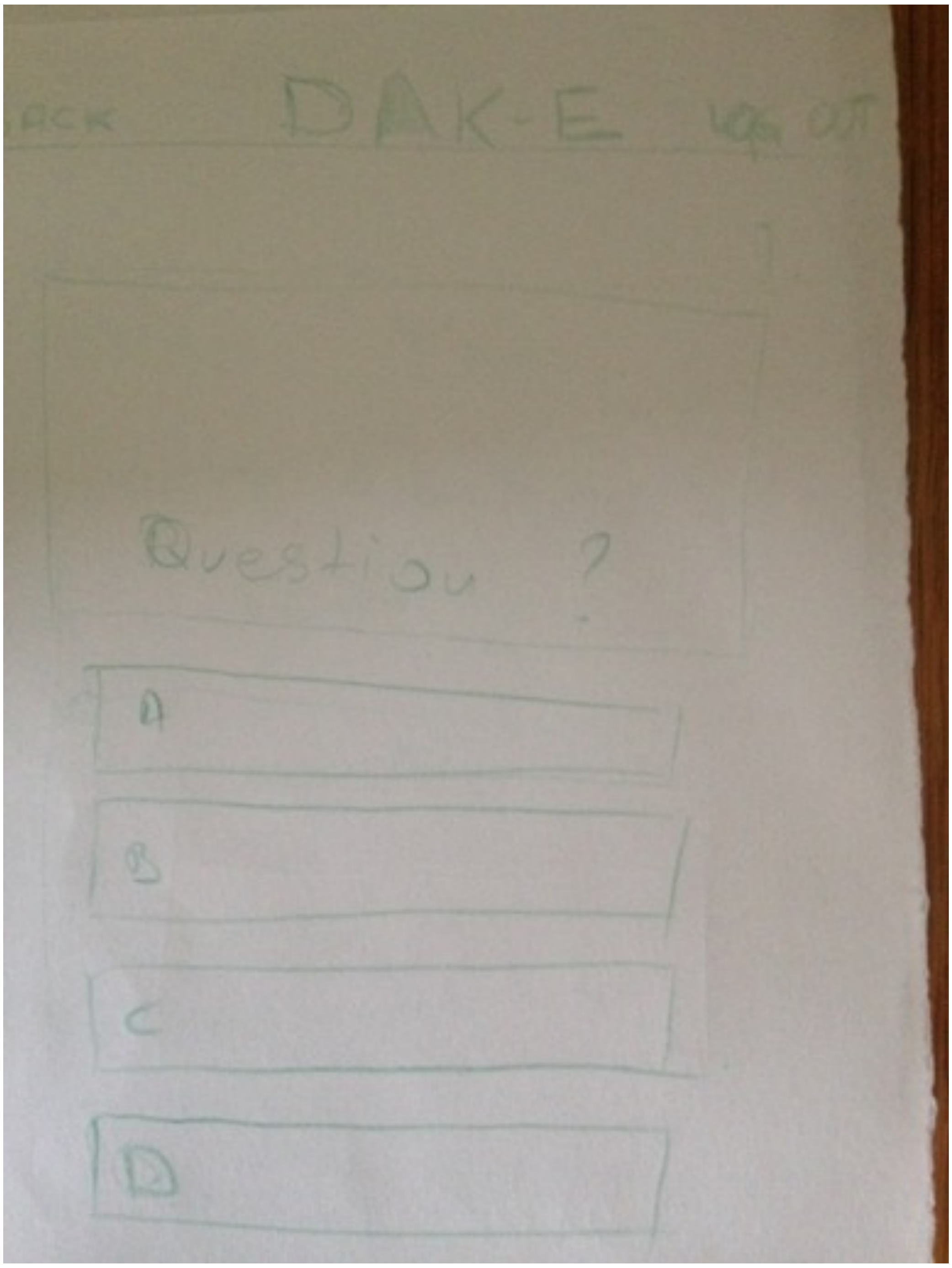
This is our second view. It shows a picture of human body. User is supposed to tap on the part of the body in which he/she feels pain. User can select few areas.



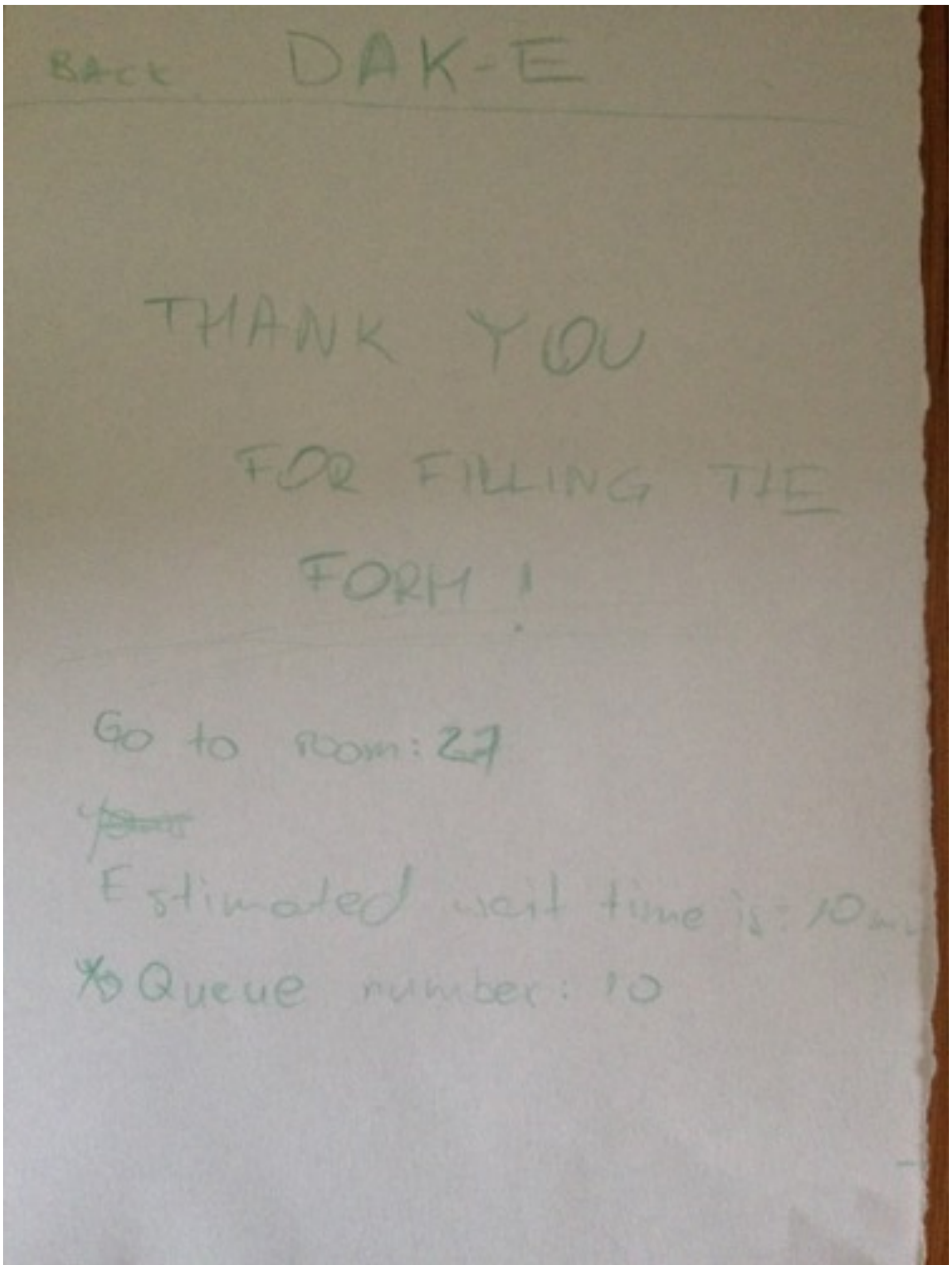
On this screen are presented most common symptoms. User should choose one or more of them. The aim of this screen is to select proper questions. We thought that it could save patient’s time, because he/she would only answer on relevant questions.



This is a question view. There is a label with question and buttons with answers. User is supposed to tap on selected answer.



This is the last view. It tells user where to go and approximately how long to wait.



BACK DAK-E

THANK YOU

FOR FILLING THE

FORM !

Go to Room: 27

Estimated wait time is: 10m

Queue number: 10

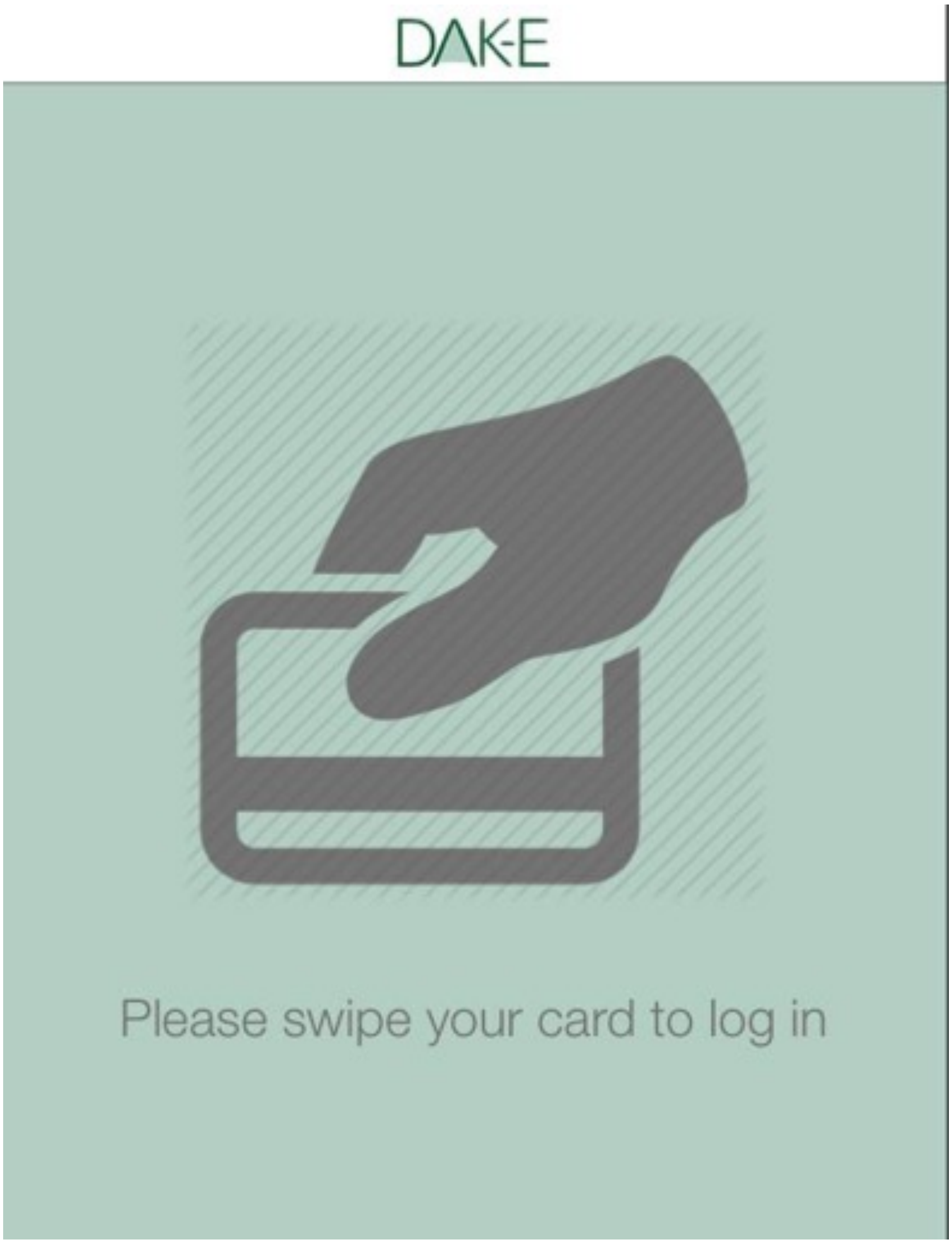
- Evaluation

  We asked some of our friends about their opinions on our idea.
  One of them commented our second screen. He told us that it is good idea but it may cause uncertainties, because there are illnesses which are hard to define. It is also possible to feel bad, but not to be able to describe it.
  Another friend told us that it would be a good idea add the ‘Next’ button instead of changing screen immediately after selecting an answer.
  After getting this feedback we decided to reject idea of showing human body and add ‘Next’ button.

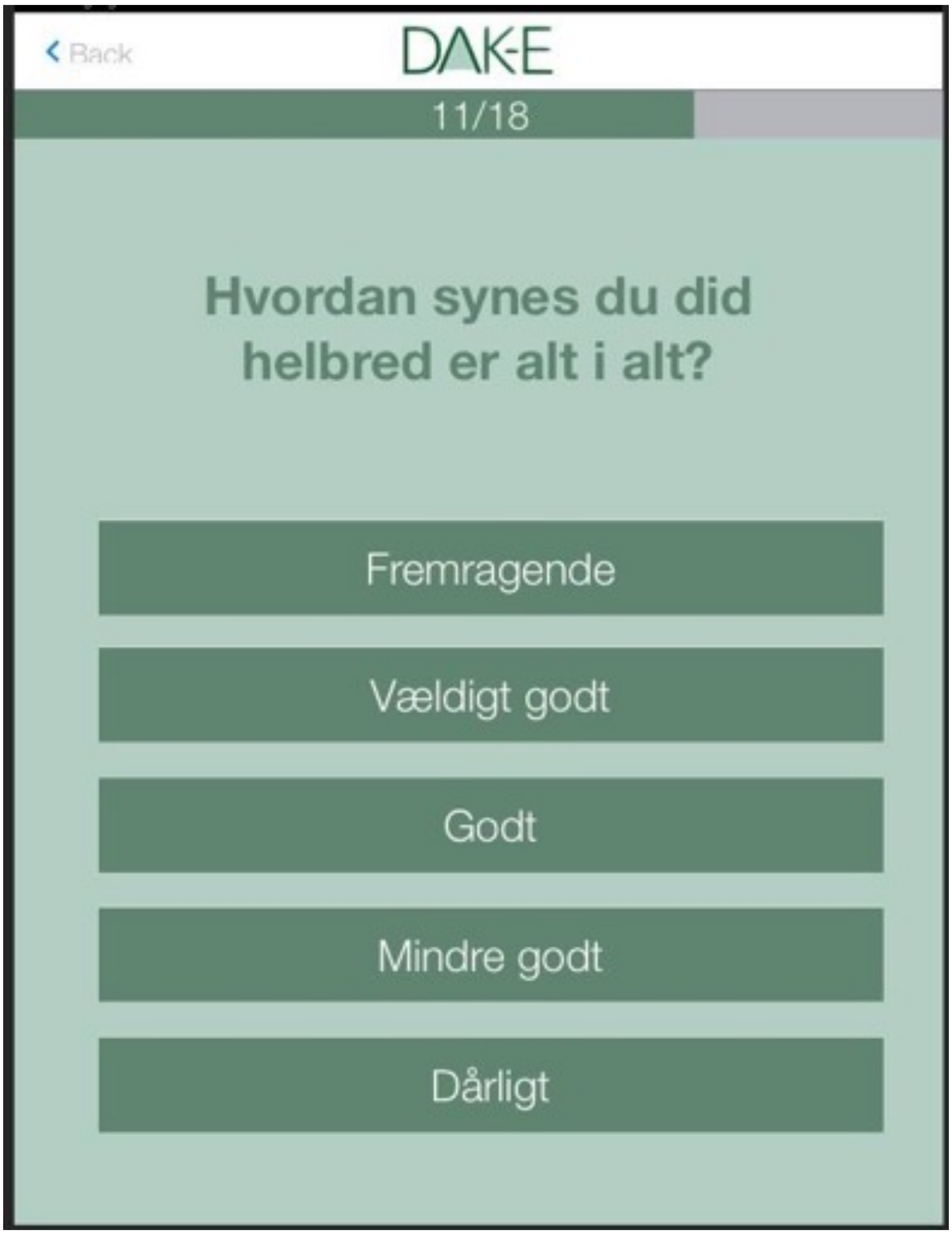
- Building the first navigational digital prototype

  We used Antetype to create digital prototype(http://www.antetype.com/). Antetype is the first design and layout software focused on visual design.
  Thanks to that we could see our final design.

  This is the first screen. Same as on the previous prototype it is possible to log in by swiping card, but there is no text field to type CPR-number.

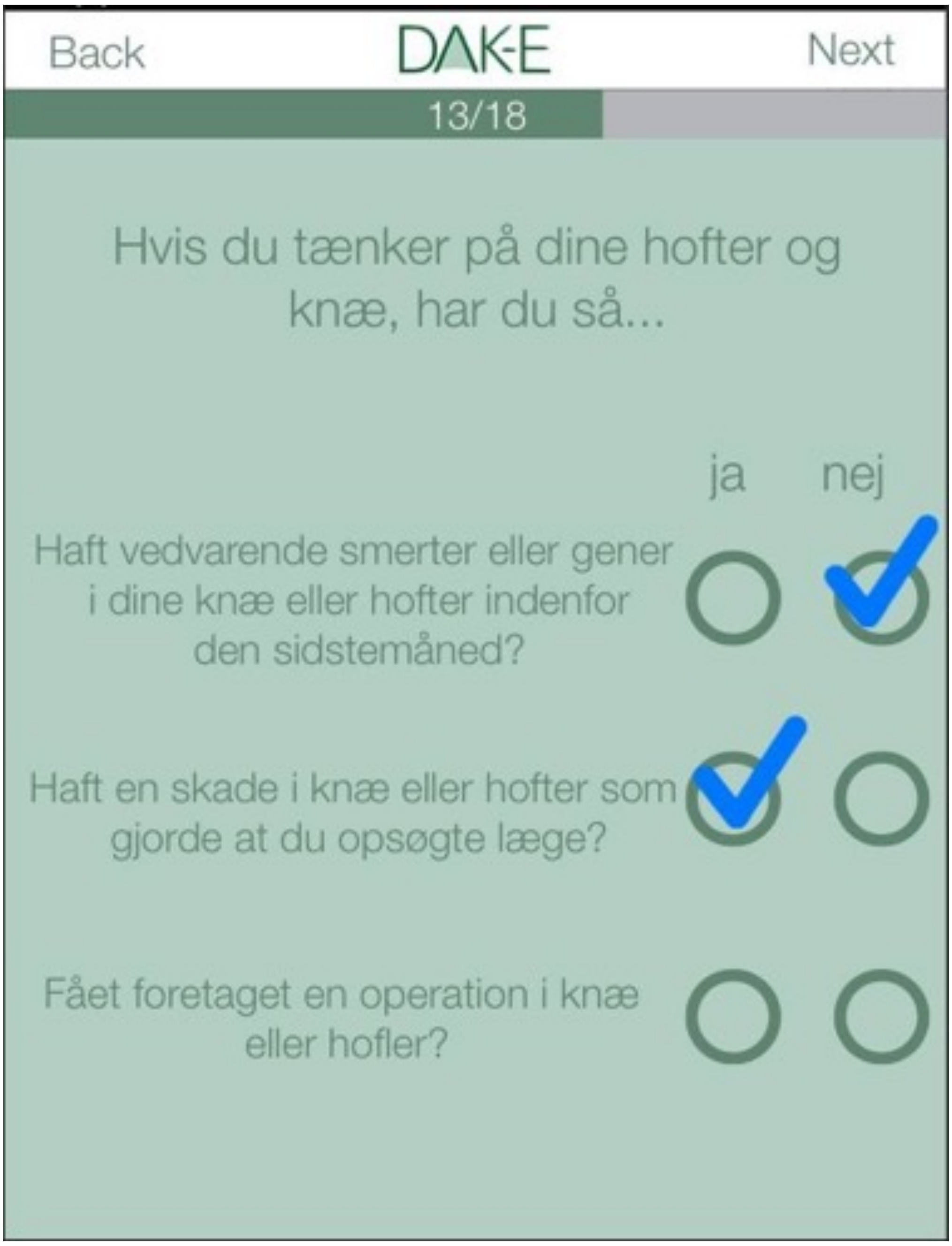


This is a question view. It looks the same as paper prototype.



Hvordan synes du did helbred er alt i alt?

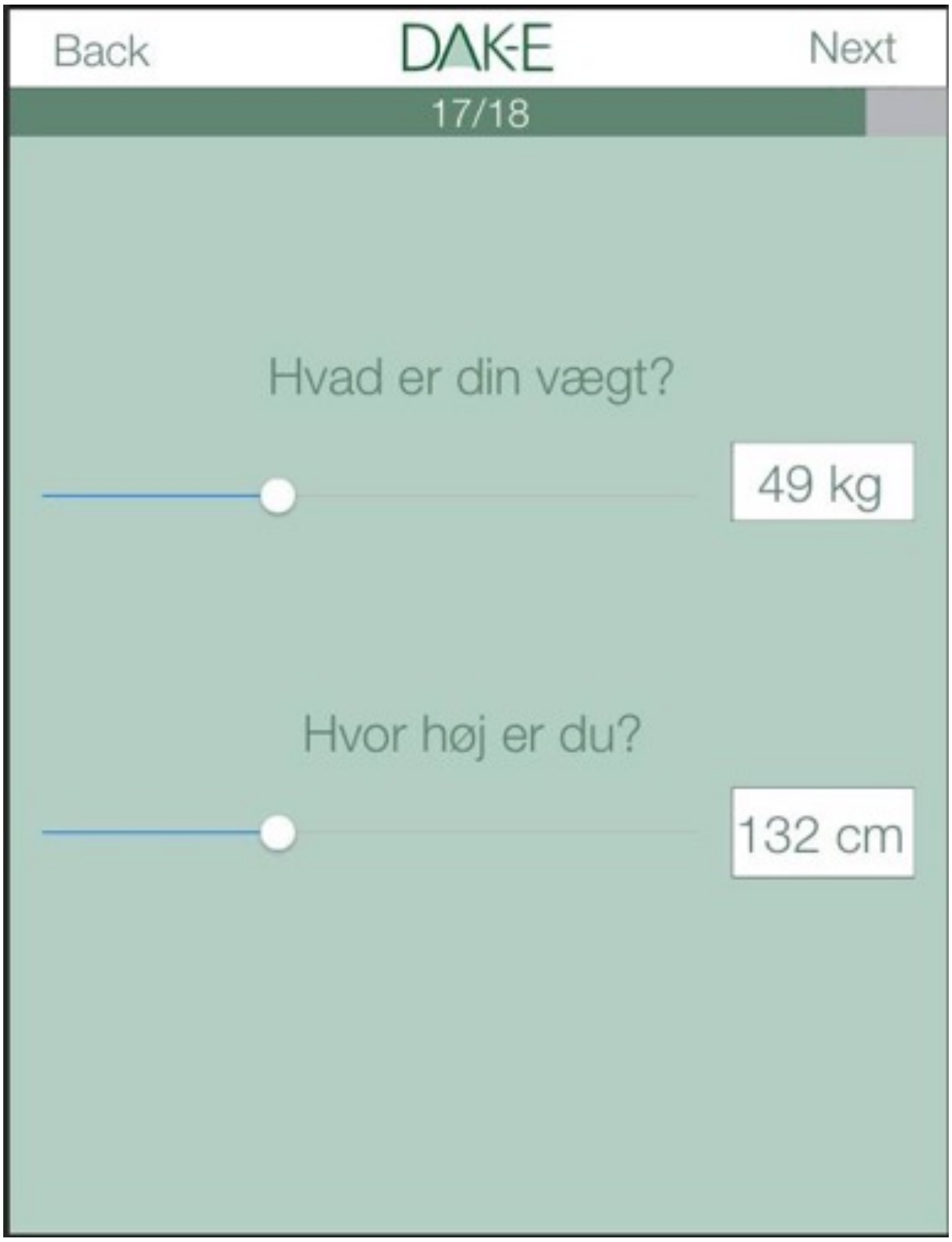
This is another question view. It contains three questions and checkboxes. User should answer all the question and then tap ‘Next’ to go to the next screen.



Hvis du tænker på dine hofter og knæ, har du så...

	ja	nej
Haft vedvarende smerter eller gener i dine knæ eller hofter indenfor den sidstemåned?	○	✓
Haft en skade i knæ eller hofter som gjorde at du opsøgte læge?	✓	○
Fået foretaget en operation i knæ eller hofler?	○	○

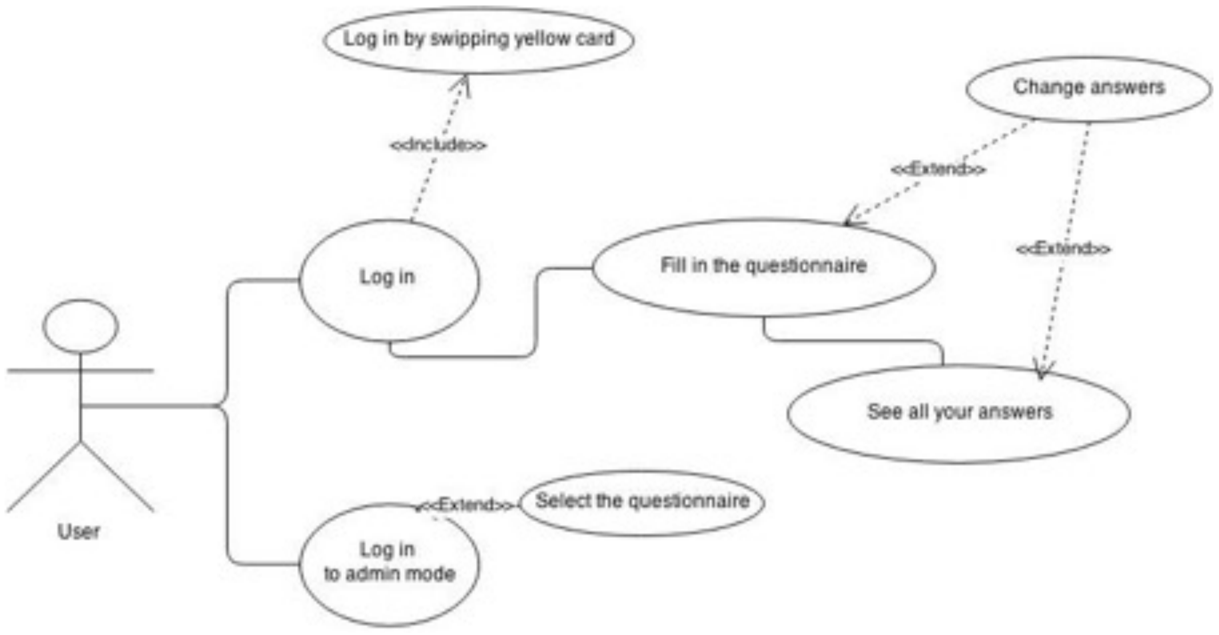
This is another question view. It shows two sliders to choose user's weight and height. User should answer both question then tap 'Next' to go to the next screen.



- Evaluation
  We met with representative of DAK-E to present him our prototypes. He told us to use bigger font's sizes, because older people may have problems with reading questions. He also suggested to create a screen on which user can see all the answers. Generally he liked our design.

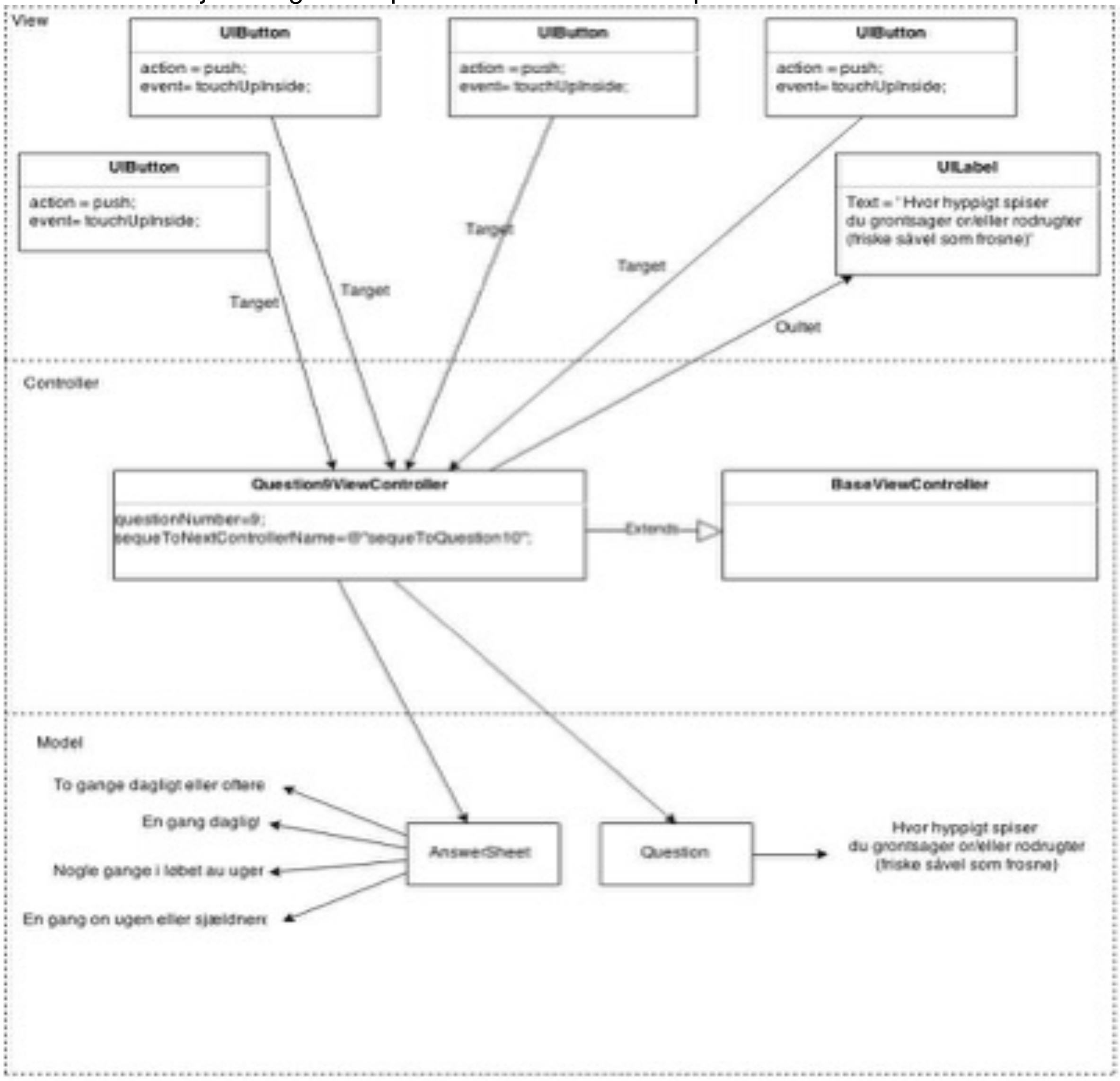
- Use-case diagram

We created a use-case diagram to present our application's functionalities.



- Object diagram

We created an object diagram to present structure of 9th question.



Finding requirements

We got first requirements form our teacher. He gave us questionnaire which he got from DAK-E. He also told us that the application would be run, because it would be convenient for patience.
He also mentioned that the most important part is questionnaire and we should not focus on the picture of human body.

Then we met with representative of DAK-E. He told us what he expected us to do. He said how the last view should look. A label in which is written Thank You. He also told us we should create view displaying all users' answers. User can go back to any question by tapping to the question. He told us create local database to store all users' answers. He told us that we should make administrative mode in which Doctor can change questionnaire or application's color. In order to change questionnaire we should implement auto-generator of views.

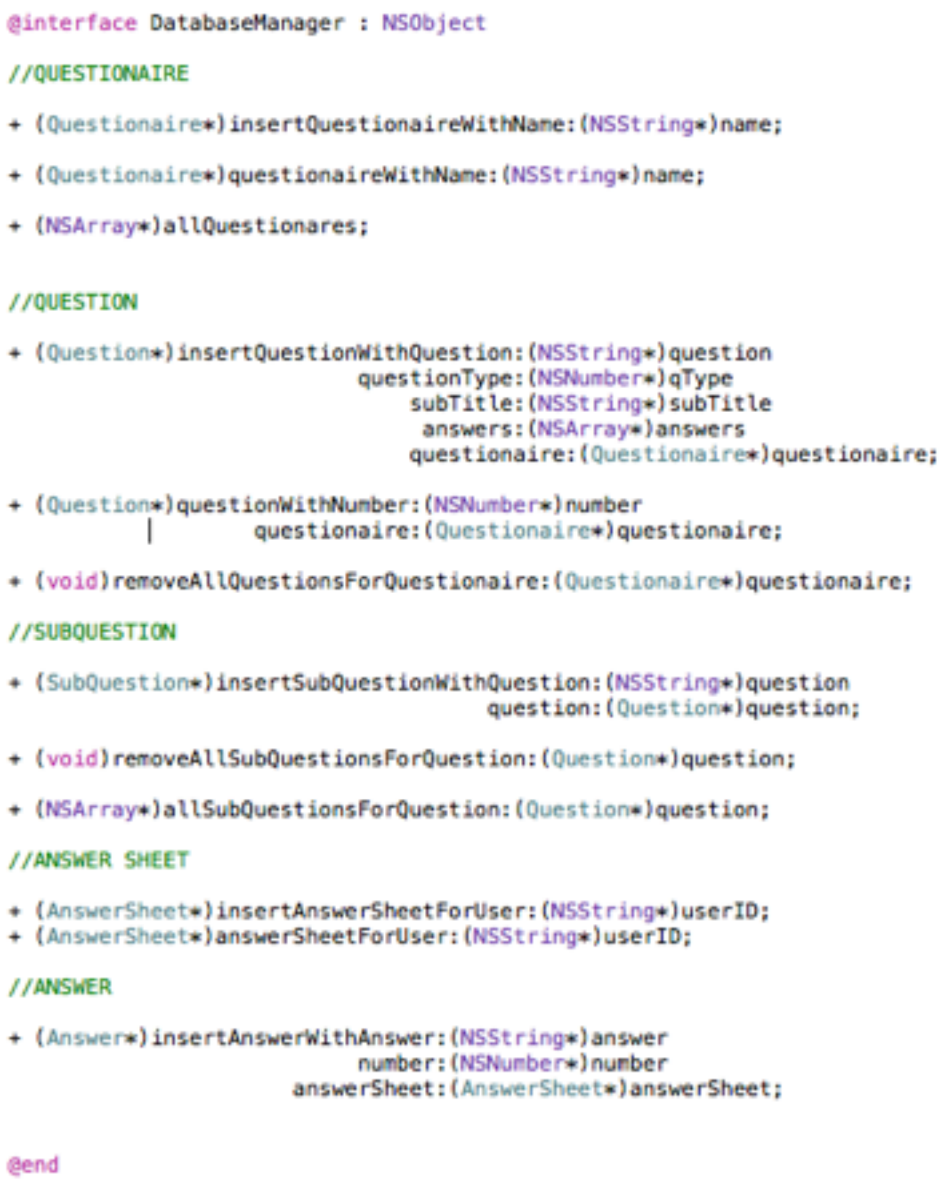
Implementation

- **Frameworks used :**

  - UIKit is an Apple framework for ios apps. We utilized UIKit to create ViewControllers, handle touch events, to add buttons, scroll views, colors, fonts, labels, slider etc. We created BaseViewController class which inherited from UIViewController class. All of our view controllers inherited from BaseViewController.
  - Foundation is a basic Apple framework in which Objective-C classes are defined. We used classes such as for example: NSString, NSArray, NSObject NSSet etc.
  - Core Data is a framework provided by Apple. We used it to manage database.

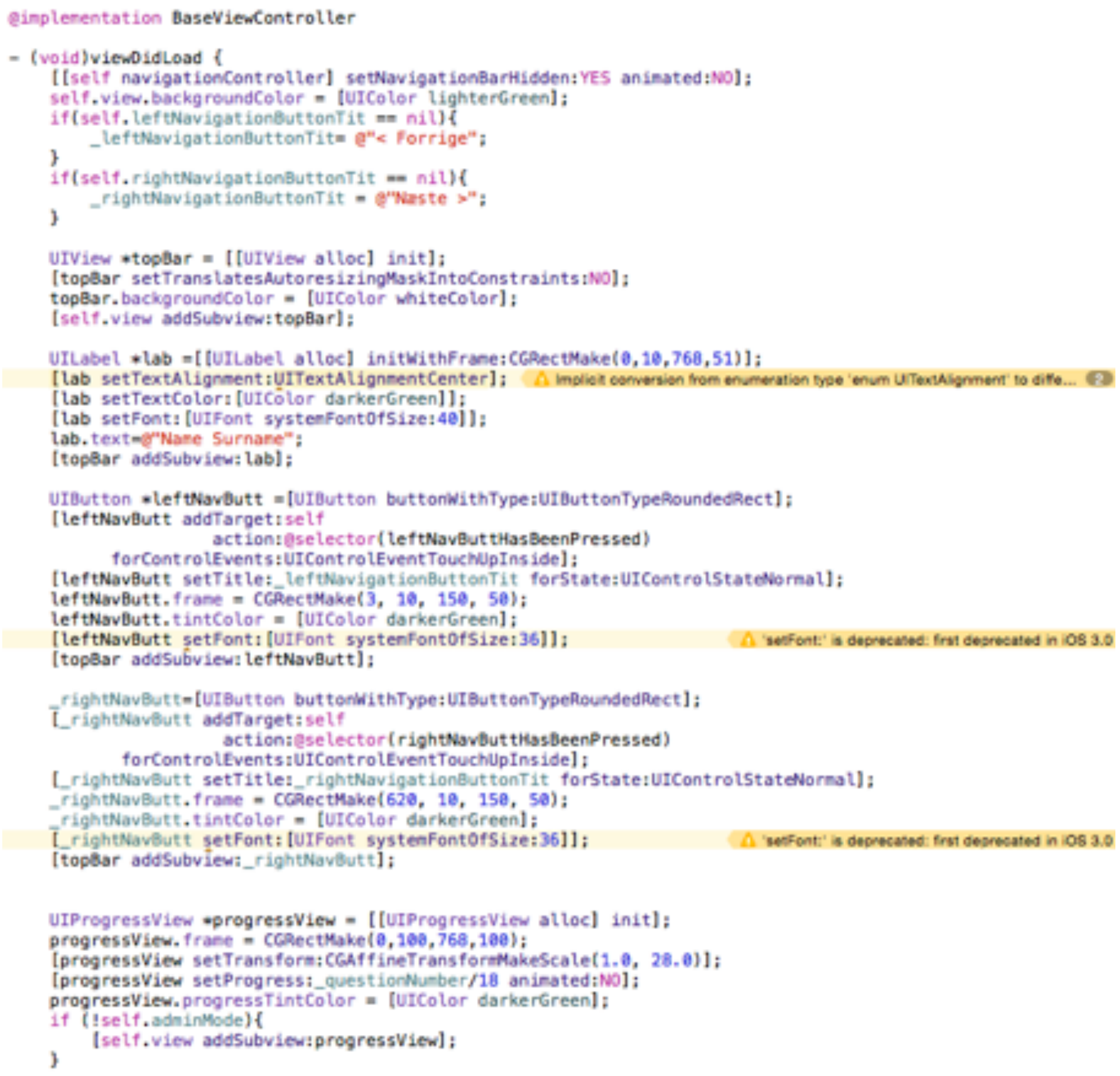
- **Classes :**

  - Model – We store data using DatabaseManager class. We hold questionnaires, question and answers.

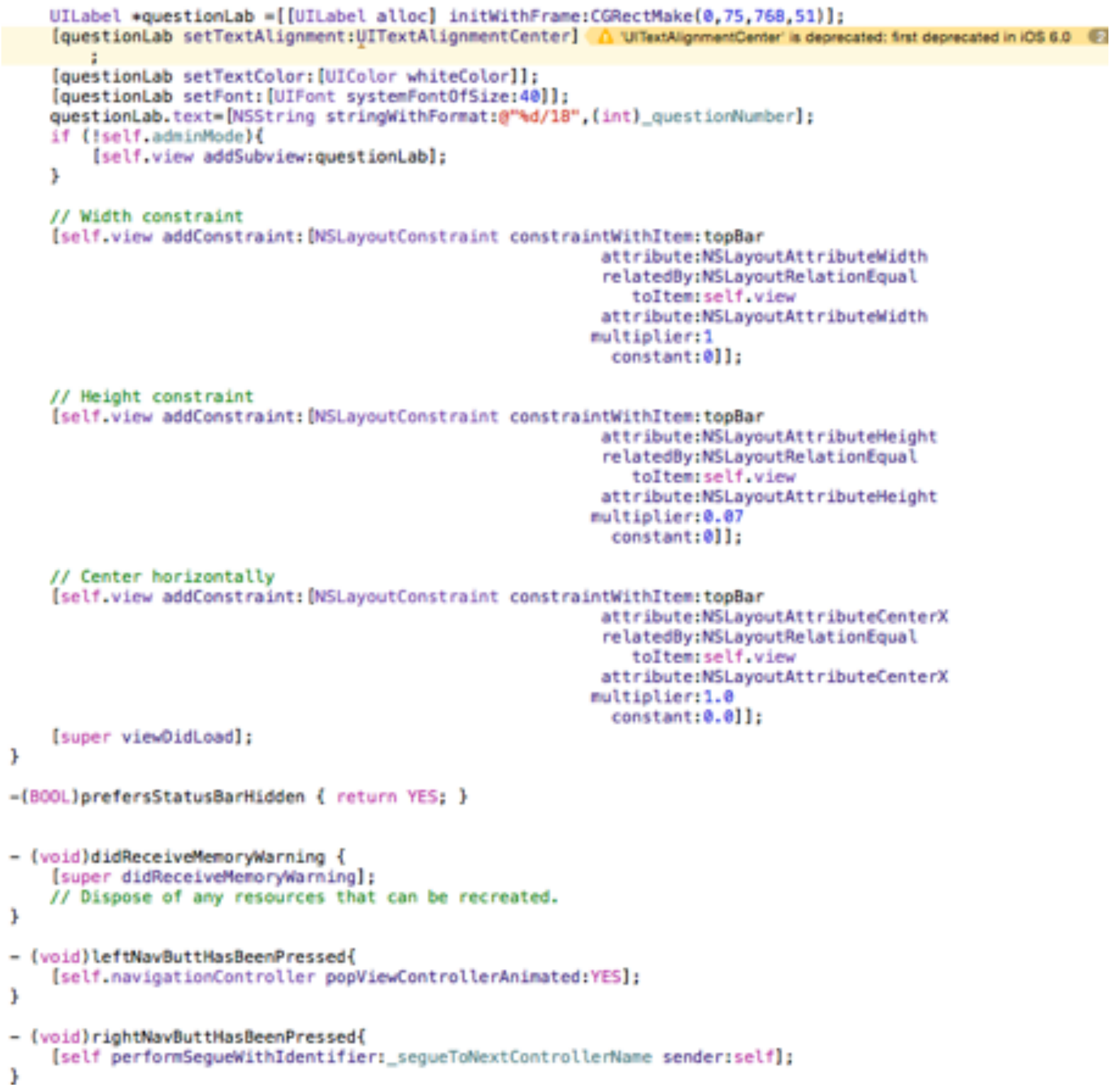


```
@interface DatabaseManager : NSObject

//QUESTIONAIRE

+ (Questionaire*)insertQuestionaireWithName:(NSString*)name;

+ (Questionaire*)questionaireWithName:(NSString*)name;

+ (NSArray*)allQuestionares;


//QUESTION

+ (Question*)insertQuestionWithQuestion:(NSString*)question
                           questionType:(NSNumber*)qType
                               subTitle:(NSString*)subTitle
                                answers:(NSArray*)answers
                            questionaire:(Questionaire*)questionaire;

+ (Question*)questionWithNumber:(NSNumber*)number
                   questionaire:(Questionaire*)questionaire;

+ (void)removeAllQuestionsForQuestionaire:(Questionaire*)questionaire;

//SUBQUESTION

+ (SubQuestion*)insertSubQuestionWithQuestion:(NSString*)question
                                     question:(Question*)question;

+ (void)removeAllSubQuestionsForQuestion:(Question*)question;

+ (NSArray*)allSubQuestionsForQuestion:(Question*)question;

//ANSWER SHEET

+ (AnswerSheet*)insertAnswerSheetForUser:(NSString*)userID;
+ (AnswerSheet*)answerSheetForUser:(NSString*)userID;

//ANSWER

+ (Answer*)insertAnswerWithAnswer:(NSString*)answer
                           number:(NSNumber*)number
                      answerSheet:(AnswerSheet*)answerSheet;


@end
```

- View – We used objects such as: buttons, labels, text fileds, slider, table view.
- Controller – We created BaseViewController class which all our view controllers inherit from.



```
@implementation BaseViewController

- (void)viewDidLoad {
    [[self navigationController] setNavigationBarHidden:YES animated:NO];
    self.view.backgroundColor = [UIColor lighterGreen];
    if(self.leftNavigationButtonTit == nil){
        _leftNavigationButtonTit= @"< Forrige";
    }
    if(self.rightNavigationButtonTit == nil){
        _rightNavigationButtonTit = @"Næste >";
    }

    UIView *topBar = [[UIView alloc] init];
    [topBar setTranslatesAutoresizingMaskIntoConstraints:NO];
    topBar.backgroundColor = [UIColor whiteColor];
    [self.view addSubview:topBar];

    UILabel *lab =[[UILabel alloc] initWithFrame:CGRectMake(0,10,768,51)];
    [lab setTextAlignment:UITextAlignmentCenter];   ⚠ Implicit conversion from enumeration type 'enum UITextAlignment' to diffe...
    [lab setTextColor:[UIColor darkerGreen]];
    [lab setFont:[UIFont systemFontOfSize:40]];
    lab.text=@"Name Surname";
    [topBar addSubview:lab];

    UIButton *leftNavButt =[UIButton buttonWithType:UIButtonTypeRoundedRect];
    [leftNavButt addTarget:self
                    action:@selector(leftNavButtHasBeenPressed)
          forControlEvents:UIControlEventTouchUpInside];
    [leftNavButt setTitle:_leftNavigationButtonTit forState:UIControlStateNormal];
    leftNavButt.frame = CGRectMake(3, 10, 150, 50);
    leftNavButt.tintColor = [UIColor darkerGreen];
    [leftNavButt setFont:[UIFont systemFontOfSize:36]];   ⚠ 'setFont:' is deprecated: first deprecated in iOS 3.0
    [topBar addSubview:leftNavButt];

    _rightNavButt=[UIButton buttonWithType:UIButtonTypeRoundedRect];
    [_rightNavButt addTarget:self
                      action:@selector(rightNavButtHasBeenPressed)
           forControlEvents:UIControlEventTouchUpInside];
    [_rightNavButt setTitle:_rightNavigationButtonTit forState:UIControlStateNormal];
    _rightNavButt.frame = CGRectMake(620, 10, 150, 50);
    _rightNavButt.tintColor = [UIColor darkerGreen];
    [_rightNavButt setFont:[UIFont systemFontOfSize:36]];   ⚠ 'setFont:' is deprecated: first deprecated in iOS 3.0
    [topBar addSubview:_rightNavButt];


    UIProgressView *progressView = [[UIProgressView alloc] init];
    progressView.frame = CGRectMake(0,100,768,100);
    [progressView setTransform:CGAffineTransformMakeScale(1.0, 28.0)];
    [progressView setProgress:_questionNumber/18 animated:NO];
    progressView.progressTintColor = [UIColor darkerGreen];
    if (!self.adminMode){
        [self.view addSubview:progressView];
    }
```



```
    UILabel *questionLab =[[UILabel alloc] initWithFrame:CGRectMake(0,75,768,51)];
    [questionLab setTextAlignment:UITextAlignmentCenter]  ⚠ 'UITextAlignmentCenter' is deprecated: first deprecated in iOS 6.0
        ;
    [questionLab setTextColor:[UIColor whiteColor]];
    [questionLab setFont:[UIFont systemFontOfSize:40]];
    questionLab.text=[NSString stringWithFormat:@"%d/18",(int)_questionNumber];
    if (!self.adminMode){
        [self.view addSubview:questionLab];
    }

    // Width constraint
    [self.view addConstraint:[NSLayoutConstraint constraintWithItem:topBar
                                                          attribute:NSLayoutAttributeWidth
                                                          relatedBy:NSLayoutRelationEqual
                                                             toItem:self.view
                                                          attribute:NSLayoutAttributeWidth
                                                         multiplier:1
                                                           constant:0]];

    // Height constraint
    [self.view addConstraint:[NSLayoutConstraint constraintWithItem:topBar
                                                          attribute:NSLayoutAttributeHeight
                                                          relatedBy:NSLayoutRelationEqual
                                                             toItem:self.view
                                                          attribute:NSLayoutAttributeHeight
                                                         multiplier:0.07
                                                           constant:0]];

    // Center horizontally
    [self.view addConstraint:[NSLayoutConstraint constraintWithItem:topBar
                                                          attribute:NSLayoutAttributeCenterX
                                                          relatedBy:NSLayoutRelationEqual
                                                             toItem:self.view
                                                          attribute:NSLayoutAttributeCenterX
                                                         multiplier:1.0
                                                           constant:0.0]];

    [super viewDidLoad];
}

-(BOOL)prefersStatusBarHidden { return YES; }


- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

- (void)leftNavButtHasBeenPressed{
    [self.navigationController popViewControllerAnimated:YES];
}

- (void)rightNavButtHasBeenPressed{
    [self performSegueWithIdentifier:_segueToNextControllerName sender:self];
}
```

Discussion

We presented our application to our group. Mostly they liked it. They told us that we should have bigger fonts. They also told us that instead of going to the next screen after choosing answer user should use answer and click next, because it may be confusing and by accident some people may choose wrong answers. They told us that it easy to fill in the questionnaire and moreover they also said that it does not take much time. They said that the design is interesting and liked the colors which we have chosen.

We reached almost all of our objectives. The most important part of our application is questionnaire and we successfully implemented it. We additionally created database in which we store user's answers. We also added administrator mode in which it is possible to choose questionnaire. Another feature is that after user finishes filling in questionnaire the application refreshed and ready for next user. Unfortunately we did not add log in by sliding yellow card because we did not get magnetic stripe card reader.
We tried make our application as simple as possible because it may be used by older people.

As stated above we need to implement log in by swiping card and changing application colors in admin mode.
As far as we know there are no similar applications.

To summarize our application is almost complete product. After adding some further features it will be able to be used in medical clinics. While developing application we learned how to use view controllers and navigate between them, to use storyboard, to manage database. We also learned how to organize team work.

To continue our project we would add missing features and test it among medical clinics patients.