Yuyang Yu

201063055

Sebastian Coope, Michele Zito

Dementia Patient Monitoring and Care Support

Abstract

Carrying for patients with dementia on their own home is very challenging and time consuming. This report demonstrates the implementation of an iOS mobile application, which is called *Dementia Care*. This application not only helps people to look after early-stage dementia patients, but also allows patients suffer less from the gradual decrease in the ability to think and remember. For caregivers, *Dementia Care* enables them to remotely monitor their patients in terms of daily routine progress and real-time location. For patients, this application could be a daily routine reminder, a memo pad, an address book, and an emergency alarm button. Technically, the implementation of this project involves several programming languages and techniques. For instance, the front-end of this application is implemented by Objective-C, while SQLite, MySQL, PHP and JSON are used for its back-end. The outcome of this project is a complete software that is ready for use. This report provides a detailed account of the progress made in creating such a software with key areas of interest such as the design and realisation of this product. Based on various evaluation criteria and assess method, an evaluation of this project is included as well.

Contents

1. Introduction

1.1. The Problem

Dementia is a broad category of brain diseases that may occur to everyone during the process of ageing [1]. Unfortunately, there is no cure for it [2]. That is to say, once people suffered from dementia, it will be a long-term molestation to both the patient and his/her family. Dementia can cause long term and often gradual decrease in the ability to think and remember that is great enough to affect a person;s daily functioning. Even worse, it is overwhelming not only for the people who have it, but also for their caregivers and families. However, the vast majority of early-stage dementia patients still want to live in their own home rather than a hospital, since it is much more convenient, private and affordable [3].

Now, here comes a number of problems. The major problem to arise is how to help dementia patients to fight with memory loss. Another issue is how to enable caregivers to monitor their patients well-fare in real-time. Alongside this, it is also important to consider having an alarming system to assist patients in coping with an emergency by contacting their caregivers immediately.

1.2. Aims and Objectives

The aim of the project is to develop a mobile based iOS application to enable people to remotely look after early-stage dementia patients who live in their own home. Moreover, by applying this project, dementia patients can have a golden chance to appreciate the subtlety and profundity of independence while they are still under kind support and care. It should also consider about contingency cases.

The objectives of the application include:

- To establish a fully functional login system with two user types (Patient or Caregiver).
  - This will enable user to sign up, first-time-only sign in, auto login, logout, and reset passwords.
  - This will be implemented with PHP and MySQL database based backend.
- To provide reminder service for the patients about events such as meal times, taking tablets or visits from carers.

  - This will not only invoke local notification to remind patients, but also enable caregivers to monitor their patients' daily routine progress.
  - This will be realised by setting local notification schedules and updating progresses to the MySQL database.
- To introduce a memorandum function for the patients.
  - This will be done by providing patients with a way to create a photo enabled memo.
- To enable users to use location services.
  - This will provide dementia patients with an address book service, which enables them to save their favourite locations and navigate themselves.
  - This will enable caregivers to track their patients' real-time locations.
- To implement an emergency alarm system.
  - This will be realised by using APNS service.
- To create a patient-caregiver pair management system.
  - This will be done by providing users with ways to send pair requests, accept/decline requests and unbind with paired users.

1.3. Challenges

This project involved a number of different areas in which I had little or no knowledge, this leads to several challenges during the implementation.

Firstly, due to my lack of knowledge in the Objective-C, I had to learn how to code in Objective-C in order to implement most functionalities I desired. This was key to the success of the software since it is supposed to be an iOS application.

Second, was a powerful back-end not only to enable the application send and retrieve data locally, but also over the Internet. Again due to lack of knowledge in Objective-C programming, I had to carry out research into programming language and tools, which would provide the functionality that can be used with Objective-C to provide local and cloud database access. Once decided how to store data, the challenge of maintaining consistency was the next step and again this required a circumspect database and algorithm design.

Thirdly, another huge challenge was pushing alerts or messages over internet to several specific devices. Since it was the first time I tried to built a networking mobile application, I needed to carry out researches to find out a way to enable multiple users to send instant messages over the distributed system.

Another challenge was bettering the interface design for dementia patient users. Since this application is designed for dementia patients as well as their caregivers, I should make the usage of the application as easy as possible. The interface should be clear enough for dementia patient users. Thus, I had to carry out research during the design stage to make sure the user experience of this application can be enjoyable.

Finally, the largest challenge throughout the project was the feasibility. I had to admit it can be a huge project with lots of possible and useful features that can be added to the system. In order to deliver the software on time, I had to identify the core features to be implemented and manage my time effectively via Gantt chart.

1.4. Solution

In order to develop the application *Dementia Care*, I had to utilise a number of technologies and programming languages. The technologies and programming languages I used all work together well and are able to realise the required features of the system.

To upload and retrieve data over the Internet, I opted for a MySQL database, due to my previous knowledge of working with this type of database. However, Objective-C cannot directly talk with MySQL database, I utilised PHP to interact with the MySQL database and fetch data in JSON syntax. Besides, I want to make *Dementia Care* become more practical to the dementia patients, so I also used a local side SQLite database to store local data, such as memos and daily routine events. That is to say, even if there is no Internet access, *Dementia Care* can still work well as a reminder application for the patients, since all the reminders are scheduled with local notification services and managed by the local SQLite database. Even better, once the Internet comes back, all the changes about the daily routine table will be updated to the cloud, which enables caregivers to track their patients' progress again.

For the emergency alarm system, I utilised the APNs (Apple Push Notification service) to push remote notifications from dementia patients to their paired caregivers. The devices are identified by their unique device token, which will be uploaded to the MySQL database. Once a patient pressed the emergency button, the application will fetch all the device tokens of his/her caregivers, and push them a remote notification, which will trigger a 10 seconds loud and annoying alarming ringtone. Moreover, there is also a short and relax ringtone indicates the alert is triggered by mistake.

In order to deliver a user-friendly interface for the patients as well as caregivers, I followed several interface design principles and pattern. I also carried researches in User Modelling and Interface issues in the medical domain.

As for the software delivery, a carefully designed Gantt chart was followed in order to complete the project on time. All the features are fully considered and discussed with my supervisors in the design stage, in order to keep the software tight and powerful.

1.5. Effectiveness and Success of Solution

The effectiveness and success of solution can be viewed in a number of ways.

From the dementia patient's perspective, this application could be a reminder, a memo book, an address book enabled map application and an emergency alarm trigger. Based on these four well-developed features, an early stage dementia patient can live in their own home with desired support and care. Therefore, the patient side of *Dementia Care* can be considered as a success.

From the caregiver's perspective, this application only provides two major functions, which are location tracking and daily routine progress checking. Although *Dementia Care* provides less functions to the caregivers, it still can be regarded as a success for the caregiver side of the application. Because the only thing caregivers wanted is to find out their patients' real-time situation.

However, to fully consider the effectiveness and success of the project, there are a lot more factors need to be considered. Factors such as the reliability, efficiency and readability of the system are all key to the effectiveness of the project. Therefore the software must be tested and evaluated before it can be deemed an effective and successful solution. Testing will be carried out and a much more detailed evaluation, of effectiveness of the project, can be found later on in this report.

2. Background

2.1. Background of Problem

As already mentioned, ageing is an inevitable course of human evolution while dementia may occur [1]. That is to say, there is a huge market for developing a dementia oriented application since everybody is a potential user. However, there is no well-known or well-developed dementia oriented application nowadays [4]. Thus, *Dementia Care* is my initial attempt to fill that gap in the market. The pain point I found is that most people still want to take care of dementia patients in their own home rather than a hospital, even though dementia can seriously influence one's daily life and also pose a significant negative impact on the patient's caregivers [3]. Because living in their own home is much more convenient, private and familiar for the patients. Additionally, it is much more affordable and economical for most families. After a closer examination, by and large, it is possible to enable early stage dementia patients to enjoy the independence in their own home, while they are still under kind support and care. The silver bullet would be a well-developed mobile based application, which could not only support patients' daily lives but also allow patients' caregivers to monitor them remotely.

2.2. Existing Solutions

Nowadays, there are a bunch of existing applications which claim them are designed for dementia patients in the Apple App Store, UK market [5]. However, they are seldom updated, and most of them are a document summary, which contains a stack of dementia associated documentation. For instances, '*Dementia Caregiver Solutions*', '*Dementia Support*', '*Dementia 101*' and '*Dementia guide for cares and care providers*'. These applications are top related to dementia in the App Store, but they only provide some research paper and doctor suggestions. After a closer examination, these application mostly are transformed from a website. That is to say, they are more like a bookmark in a browser and do not provide similar functions as the project proposed.

By taking a deeper look at the current dementia related applications, I found out there is no integral and customised application for dementia patients [6]. Most of these applications are not customised for dementia patients or their caregivers, they are individual applications for

memory and focus, enjoyment, relaxation, wandering or tracking, medication management, and alzheimer's information and resources reading. Although these applications are quite helpful and powerful, they are not targeting at dementia patients. Perhaps, most people believe that old people seldom using smart phones and thus there is no need to develop a specialized application for the aged dementia patients. However, in the long run, the young people who could not live without their phones will eventually become the old. Hence, the solution produced by this project is focused on the long term.

As already described, the final application will work as a reminder, a memorandum, a map application with address book, an emergency contact tool, a location tracker and a daily routine monitor. Although there are existing similar solutions in the market, the solution of this project takes these approaches a step further, by integrating them into one system.

2.3. Reading and Research

In implementing the project, I inevitably run up against a number of difficulties. The first challenge is programming in Objective-C. As I already stated I only had little knowledge in this programming language, therefore I had to carry out a lot of reading and practicing to ensure I can use it as I wanted. This learning progress continues throughout the entire project realisation.

Once I had gained sufficient knowledge about the Objective-C programming, I then had to research and learn a number of backend and web technologies, such as MySQL, SQLite, PHP and JSON, since these skills are necessary to the project's data storing and retrieving. Also, I needed to make sure these technologies and programming languages can work together as I wanted.

Thirdly, I carefully studied how to use the Apple Push Notification service (APNs) to enable patients to instantly send emergency alerts to caregivers. This is a robust and highly efficient service for propagating remote notifications to iOS devices [7]. The reading and research involved how to automatically generate unique device tokens for each devices and then push

messages to them. Additionally, I also figured out how to define the title, content, badge number and notification ringtone of each remote notifications.

Furthermore, I also carried out a number of reading and researches into user interface design to deliver a user-friendly interface. For example, I employed the traditional 'Tab bar menus' as the navigation design style. Because the simple tap interaction is much more easier and clearer than that of swipe menu style, even if the tab bar is fixed on the screen, which means it occupies a certain amount of screen space all the time [8].

I also had to ensure I was managing my time effectively, because there are a large number of researches needed to be carried out during the implementation of *Dementia Care*.

2.4. Project Requirements

This project is target at two kind of users. Thus, the project requirements can be categorised by types of users.

Dementia Patients:

The application should support early stage dementia patients to live alone in their own home, thus the requirements include the following:

- The ability to sign up, first-time-only sign in, auto-login, log out and reset passwords.
- The ability to bind/unbind with caregivers.
- The ability to create/edit/delete daily routine reminders and be reminded as scheduled.
- The ability to create/edit/delete memos and insert pictures into the memo by taking a photo or selecting from system camera roll.
- The ability to store map annotations into an address book.
- The ability to find their location in map and check location annotations from address book.
- The ability to send emergency alerts to their caregivers.
- The ability to send an alert cancellation message to their caregivers.

Caregivers:

The system should enable caregivers to remotely monitor their patients in terms of daily routine progresses and real time locations. Hence, the requirements include the following:

- The ability to sign up, first-time-only sign in, auto-login, log out and reset passwords.
- The ability to bind/unbind with patients.
- The ability to check his/her patients' daily routine progress.
- The ability to check his/her patients' real-time locations.
- The ability to receive emergency alerts as well as cancellation messages.

3. Data Required

3.1. Data Usage

The data required in this project were simulated data, such as daily routine events, memos, locations and emergency events. It followed the University Policy on ethical use of human data and human participants.

3.2. Human Participants

I used myself as a testing participants to test different functions of *Dementia Care*, such as locating and reminding. The project exactly followed the University Policy on ethical use of human data and human participants.

3.3. Ethical Issue In Future Employment

Currently, the data usage and human participants are used as above mentioned. However, if the application is put into markets, there will be real data and other human participants involved. The data protection will be considered. All the data transmission will be encrypted. The protection of personal information will be the first priority.

4. Design

4.1. Use-Case Diagram

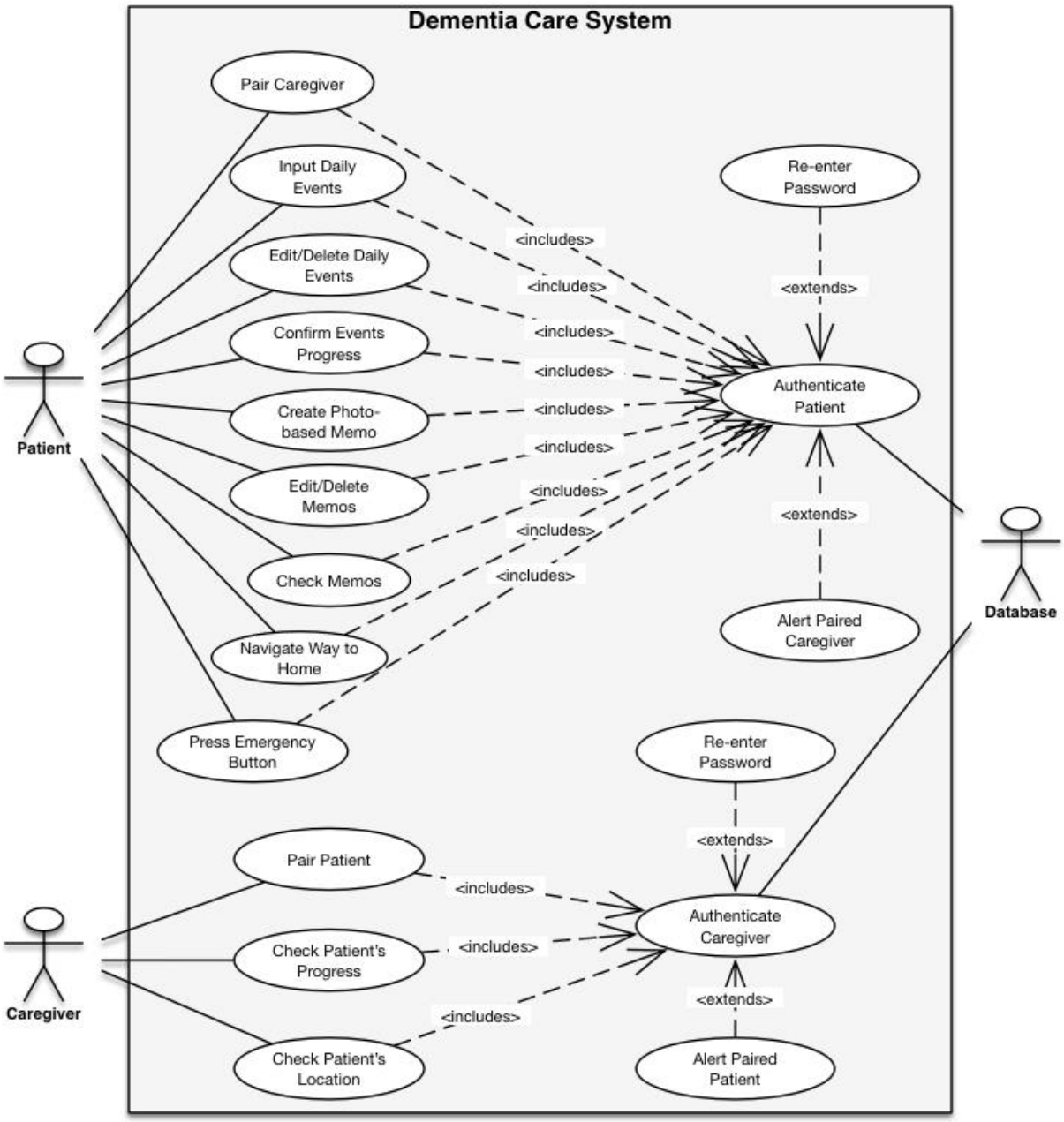

Figure 1. Use-case Diagram

4.2. Interaction Chart

4.2.1. Login

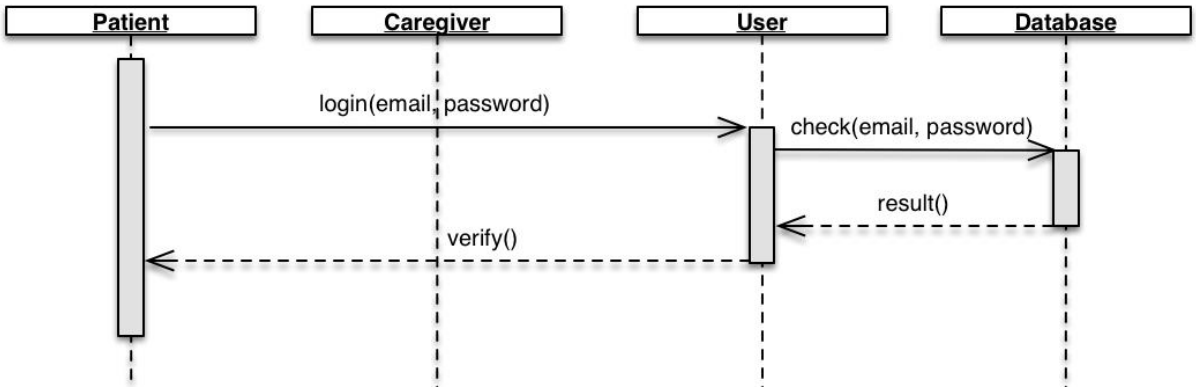

Figure 2. Login Use Case Sequence Diagram

4.2.2. Edit Daily Event

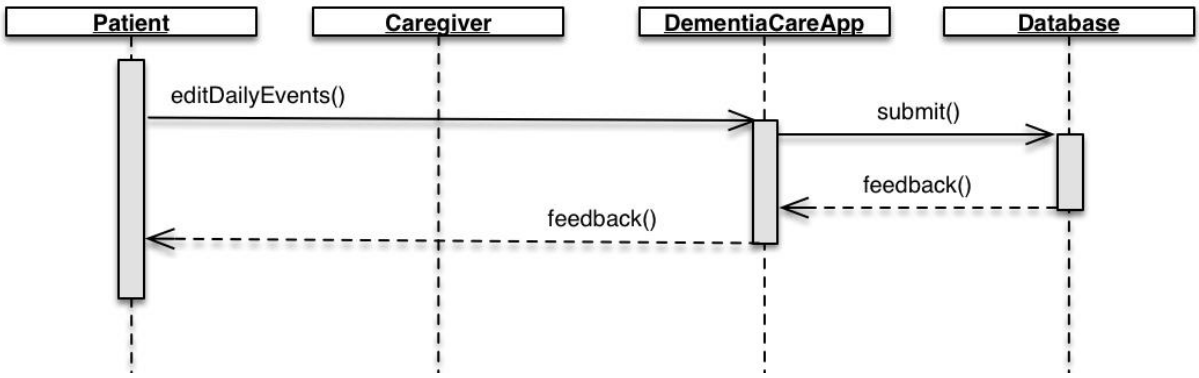

Figure 3. Edit Daily Event Sequence Diagram

4.2.3. Confirm Event Progress

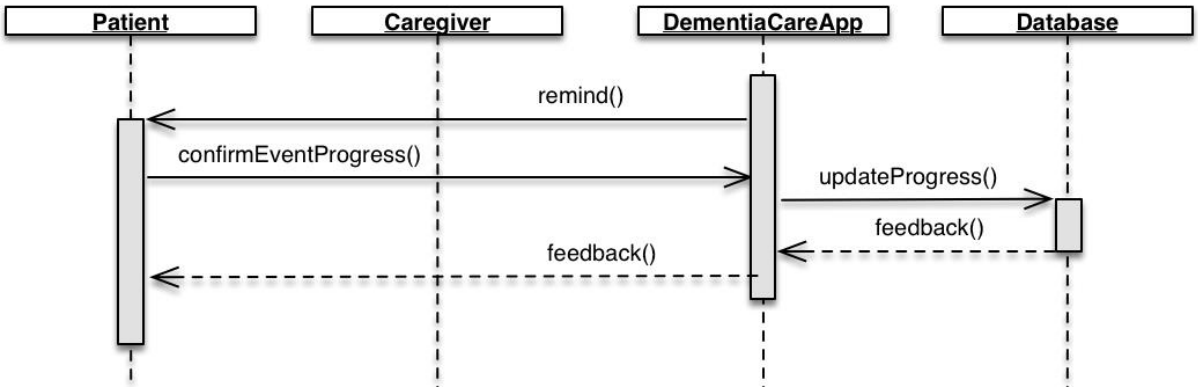

Figure 4. Confirm Event Progress Sequence Diagram

4.2.4. Update Location

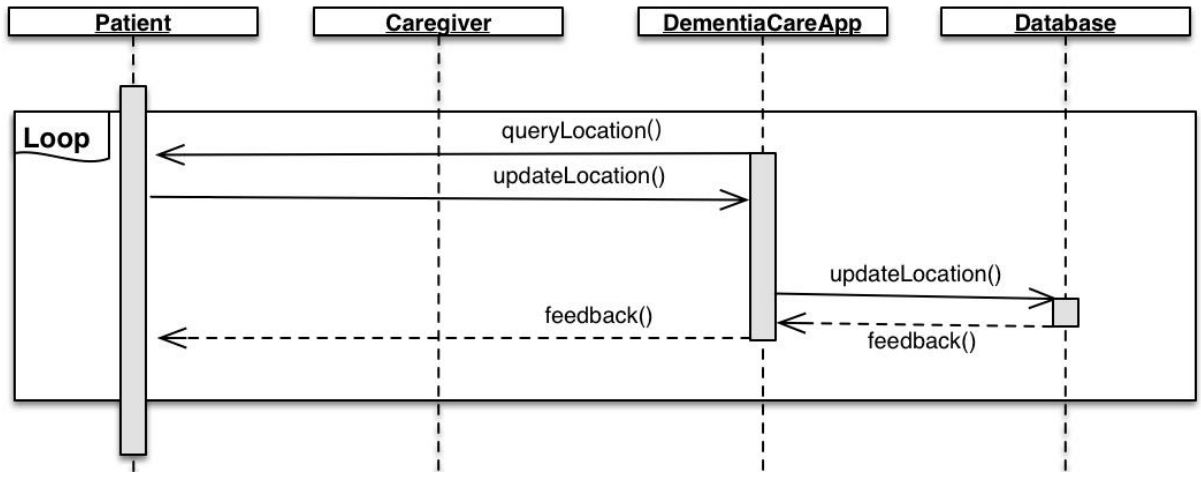

Figure 5. Update Location Interaction Chart

4.2.5. Check Patient's Progress

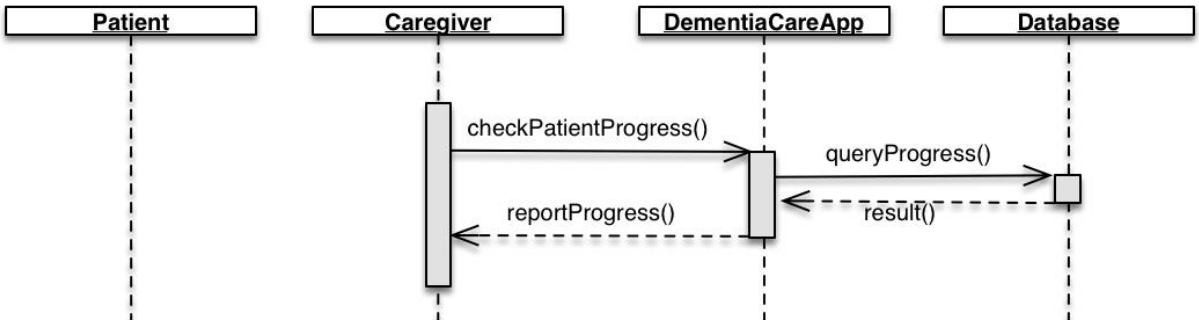

Figure 6. Check Patient's Progress Sequence Diagram

4.2.6. Press Emergency Button

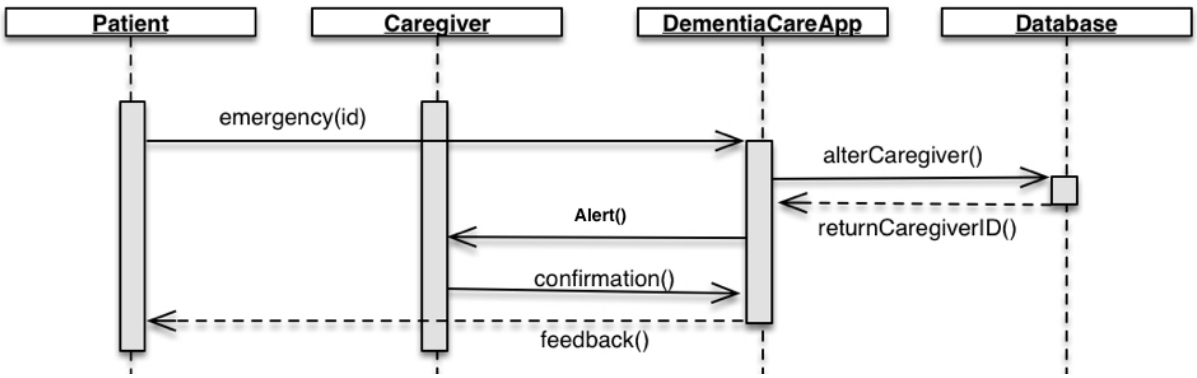

Figure 7. Press Emergency Button Sequence Diagram

4.3. Object Classes

4.3.1. CRC Cards

During the object class design stage, CRC cards were used as a brainstorming tool to design the system. For example:

User	
login logout change password set age set gender set name set user type get name get age get gender get user type	Patient Caregiver

Figure 8. User CRC card

Patient	
pair caregiver create daily event edit/delete daily event confirm event progress locate home create memo edit/delete memo check memo call emergency dis-pair knows associated caregiver	User Caregiver

Figure 9. Patient CRC card

Memo	
set title set photo set description get title get photo get description	Patient

Figure 10. Memo CRC card

4.3.2. Class Diagram

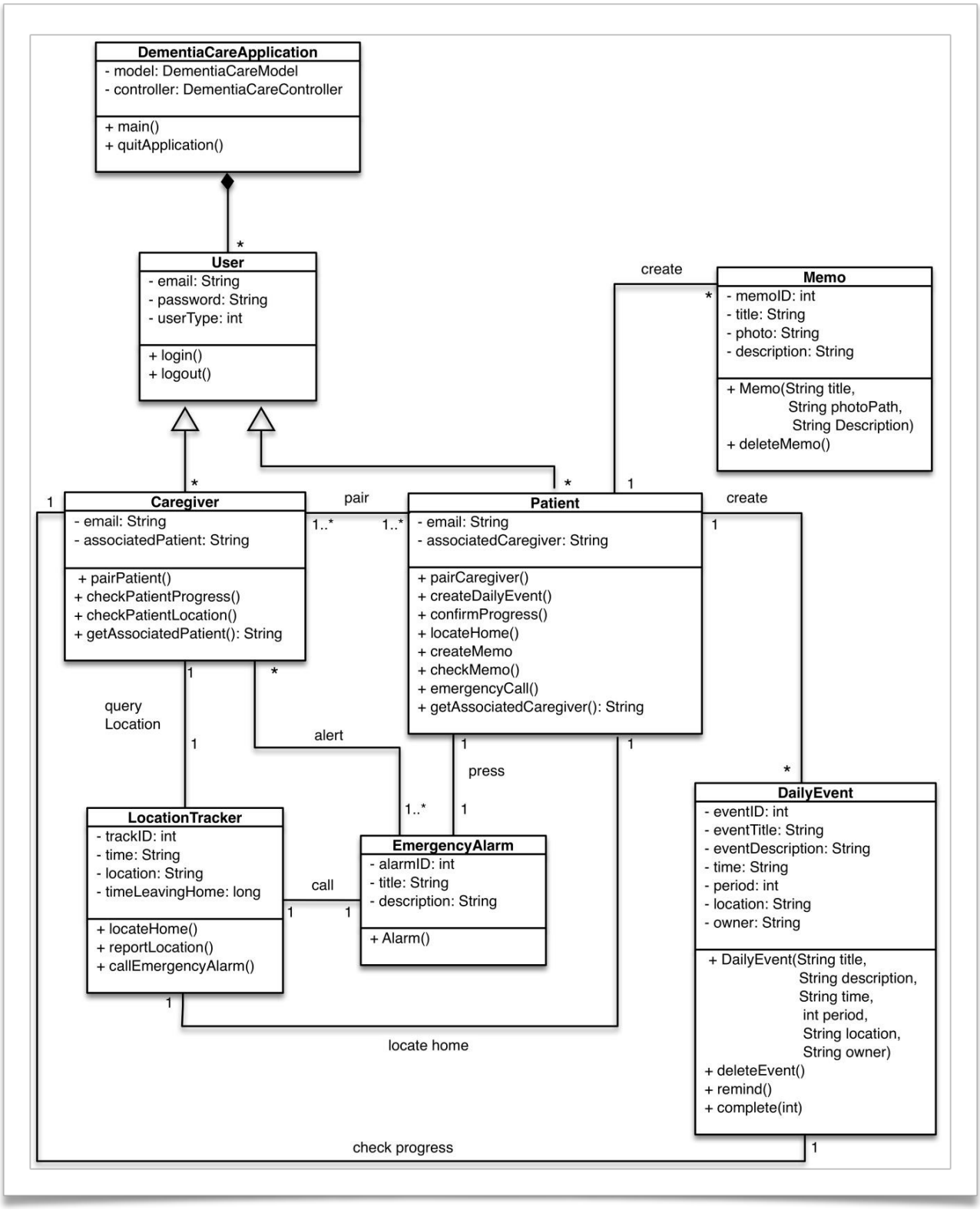

Figure 11. Class Diagram

This diagram illustrates the general class interaction and functions in this system.

4.3.3. Model View Control Diagram for iOS

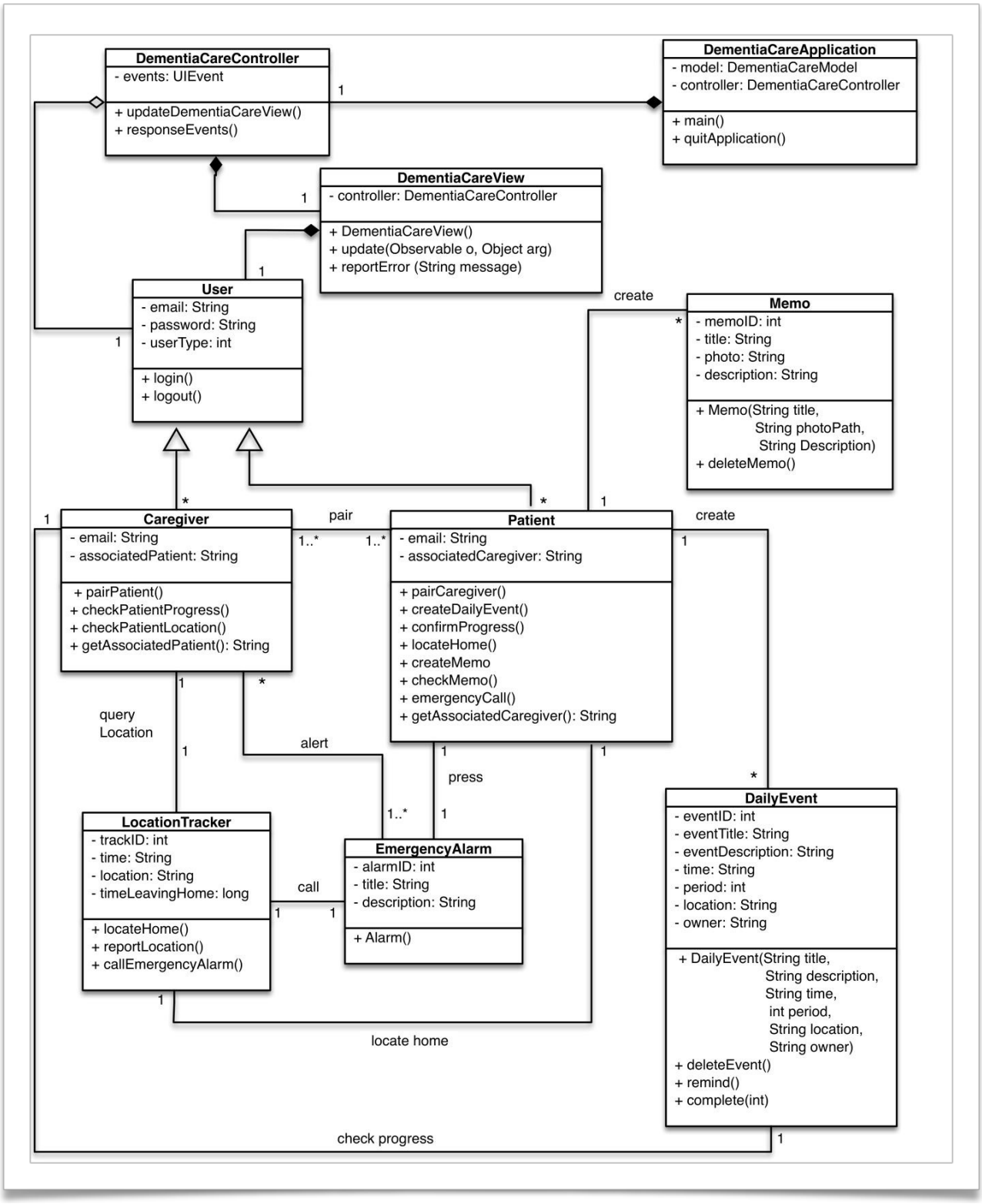

Figure 12. MVC Diagram for iOS

As shown, there are all the class functions as well as their relationships in the system for iOS application.

4.3.4. Class Descriptions

- **User**

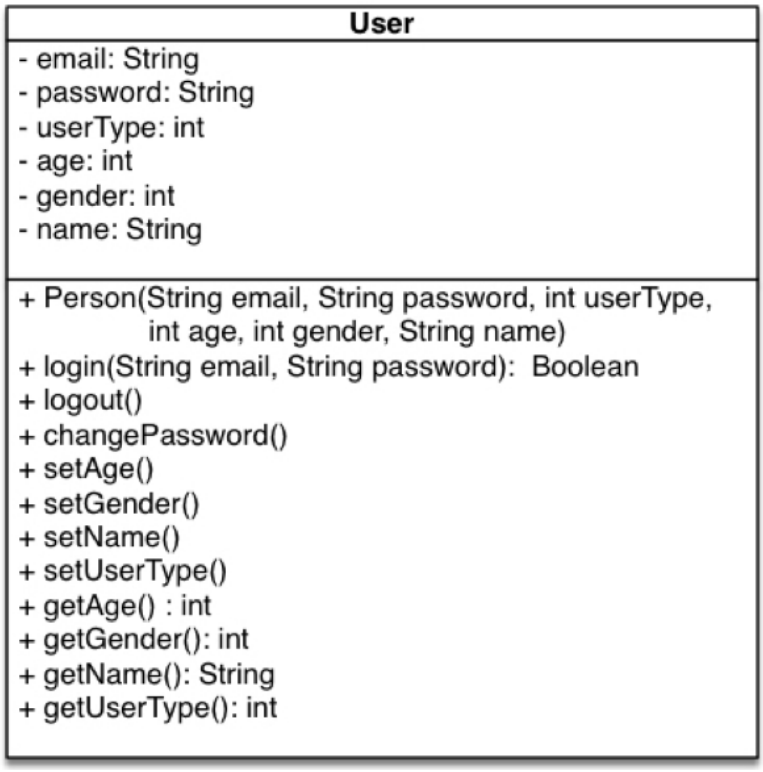

Figure 13. User Class

- **Patient**

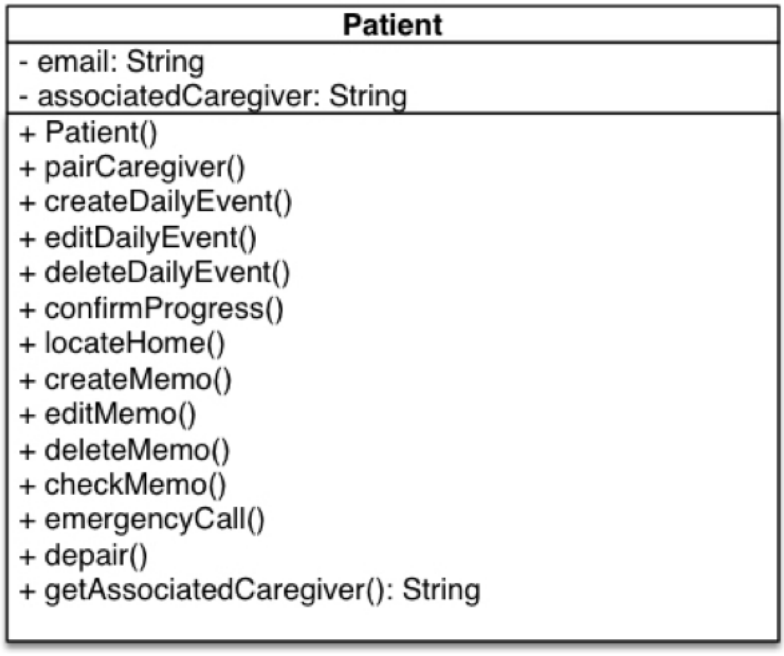

Figure 14. Patient Class

- **Caregiver**

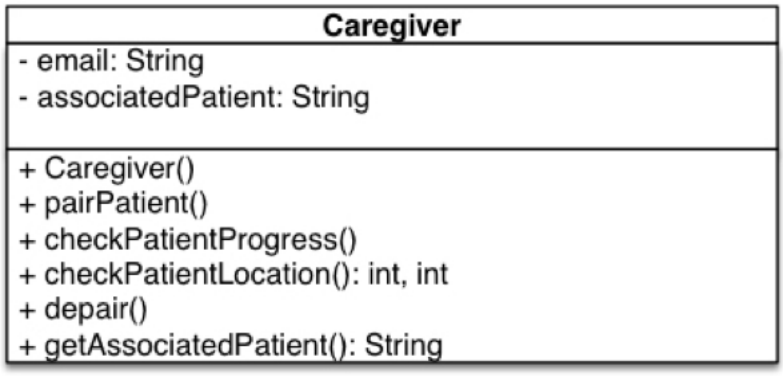

Figure 15. Caregiver Class

- **DailyEvent**

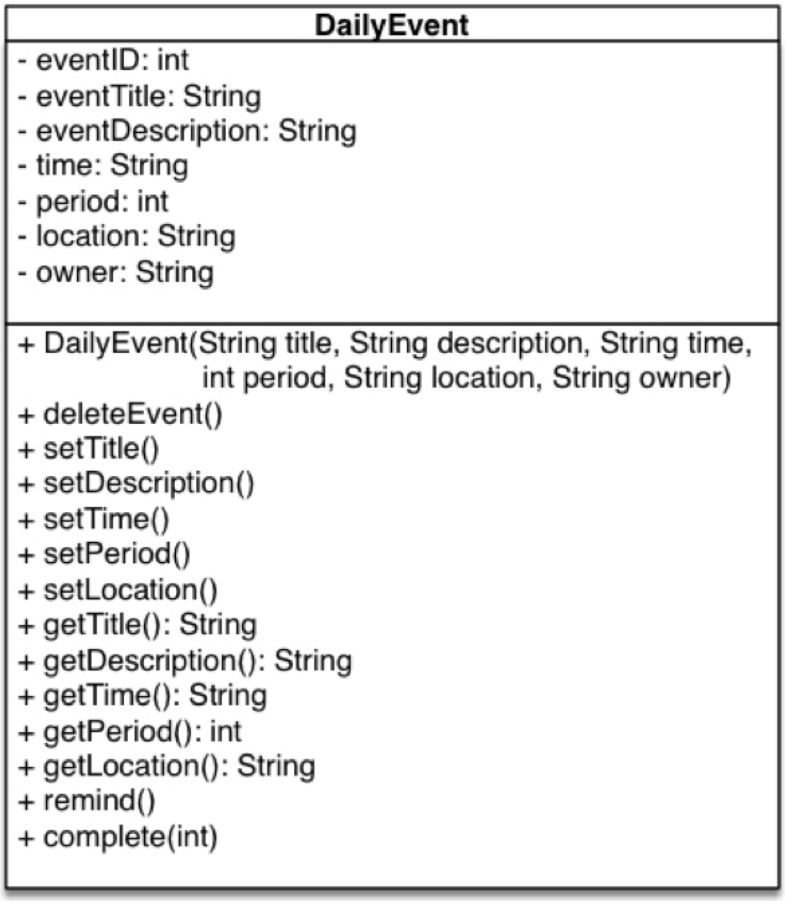

Figure 16. Daily Event Class

- **Memo**

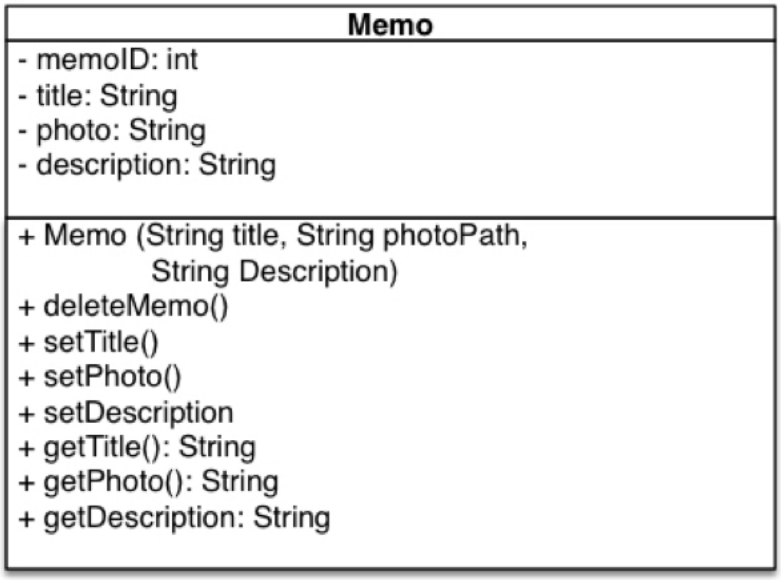

Figure 17. Memo Class

- **LocationTracker**

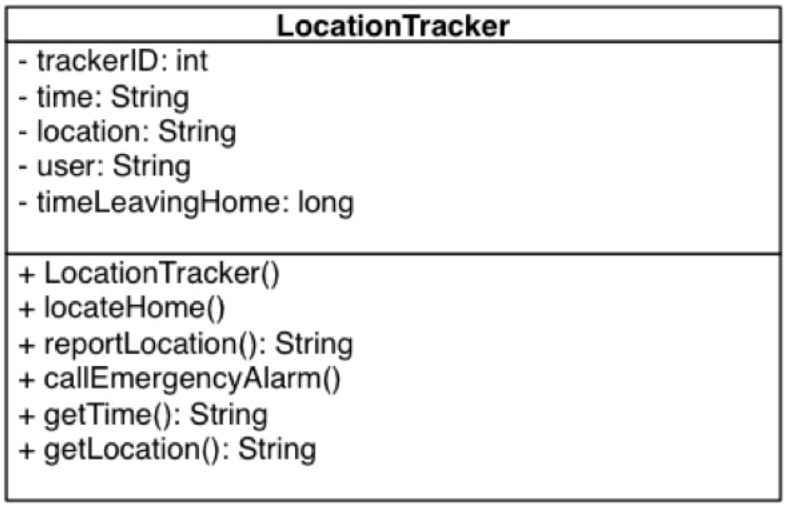

Figure 18. Location Tracker Class

- **EmergencyAlarm**

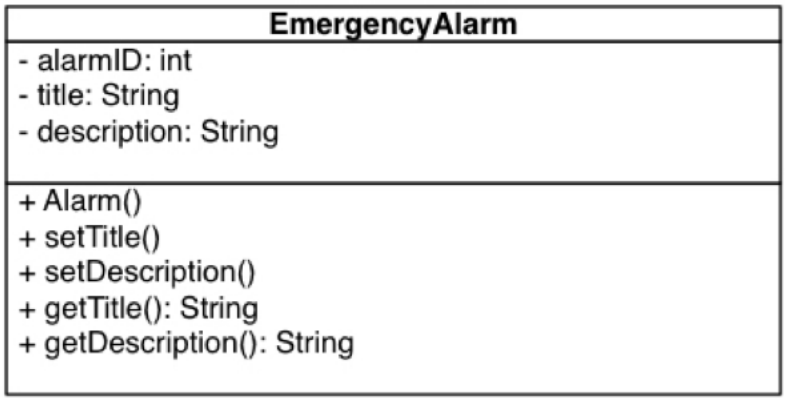

Figure 19. Emergency Alarm Class

- **DementiaCareView**

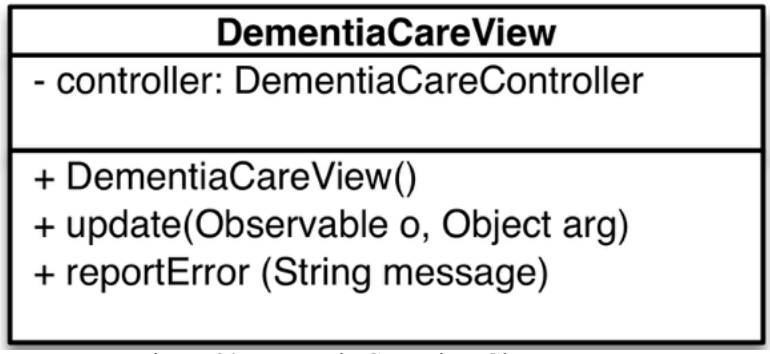

Figure 20. Dementia Care View Class

- **DementiaCareController**

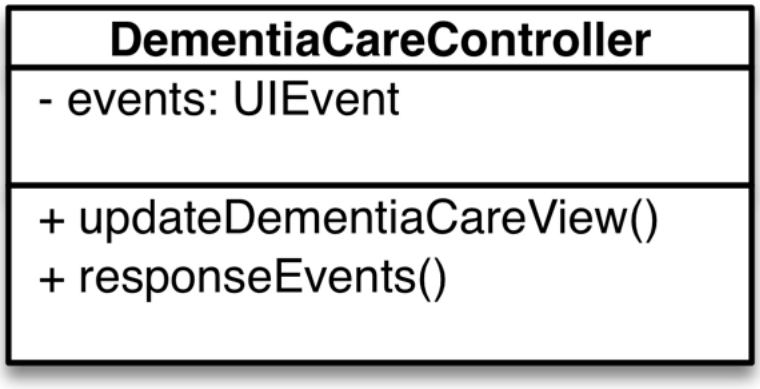

Figure 21. Dementia Care Controller Class

- **DementiaCareApplication**

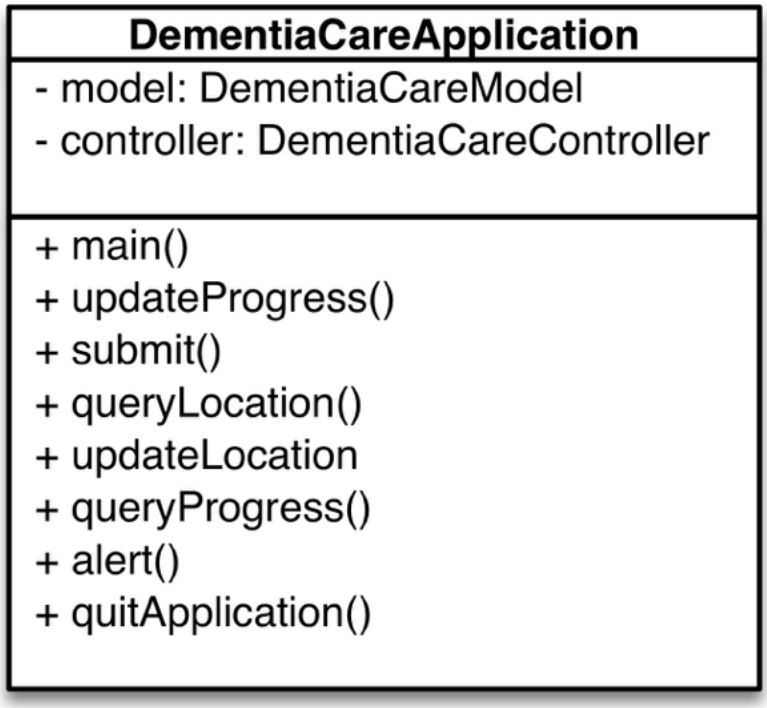

Figure 22. Dementia Care Application Class

4.4. Pseudo-code for Key Methods

- **Login**

```
// Log in. Check password, if attempted over 5 times, alert linked user.
- (BOOL) login:(NSString *)email andPassword: (NSString *)pswd {
	if (pswd == getPasswordFromDB(email) {
		output: login success!
		if (user_type == patient)
			jump to patient_homePage;
		else
			jump to caregiver_homePage;
	}
	else {
		output: wrong password! Please re-enter.
		if (reenter_times >= 5)
			alert associated_user;
	}
```

- **Pair associated user**

```
// Send a pair request to specified user.
- (void) pair:(NSString *)target_email {
	if (currentUserType != getUserType(target_email)) {
		if (pairlist.has(target_email) == null)
			sendRequest(current_email);
		else
			output: Sorry, he/she is already paired with you!
	}
	else if (currentUserType == patient)
		output: Sorry, you need to pair with caregivers.
	else
		output: Sorry, you need to pair with patients.
}

// Handle pair requests.
- (void) handleRequest:(NSString *) request_email {
	display request;
	if (click Agree_button) {
		pairlist.add(request_email);
		replyDecision(1, current_email);
	}
	if (click Decline_button) {
		replyDecision(0, current_email);
	}
}
```

```
// Handle Answers. Add the user into pair list if he/she agreed.
- (void) handleReply:(NSInteger) reply andFrom: (NSString *) email {
      if (reply == 0)     // decline
            output: Sorry, getName(email) declined your request.
      if (reply == 1) {
            pairlist.add(email);
            output: Congratulation! You just paired with getName(email);
      }
      else
            errorReport;
}
```

- **Daily event**

```
// Create Daily event and upload it to the database.
- (void) setDailyEvent:(String)title andDescription: (String)description
                                          andTime: (String)time
                                          andPeriod: (Int)period
                                          andLocation: (String)location {
      new DailyEvent(description, time, period, location, owner);
      upload Event;
      set localRemindTime;
}

// Remind patient according to DailyEvent and wait for confirmation.
- (void) reminder {
      call system_alarm;
      display event;
      if (click Complete_button) {
            update(user_email, event_id, 1)
            output Good Job.
      }
      if (click Cancel_button) {
            update(user_email, event_id, 0)
            output This event has been cancelled.
      }
      else {       // handle error situation
            update(user_email, event_id, 2)
            output Error occurs.
      }
}
```

```
// Update progress in database, in order to be checked by caregivers.
- (void) update:(NSString *)e forEvent:(NSInteger)id
                                  andStatus:(NSInteger)s {
      open socket;
      select database;
      DailyEvents SET Status=s WHERE Email=e AND Event_id=id
      output feedback;
      close socket;
}


// Check patient's progress of daily events.
- (void) check:(NSString *)email {
      open socket;
      select database;
      SELECT Event_title, Status FROM DailyEvents WHERE Owner=email
      output result;
      close socket;
}
```

- **Emergency Alarm**

```
// Patient press emergency alarm.
- (void) emergencyAlarm:(UIEvent *)pressEmergencyButton {
      for (i = 0; i < pairlist.size, i++) {
            open socket(pairlist(i));
            send alert(getCurrentEmail);
            close socket;
      }
}

// Caregiver receive alarm and handle it.
- (void) getAlarm(NSString *)patient {
      if (click Confirm_Button) {
            open socket(patient);
            send alarm_recieved;
            close socket;
      }
}
```

4.5. Interface Design

- **Login Interface**

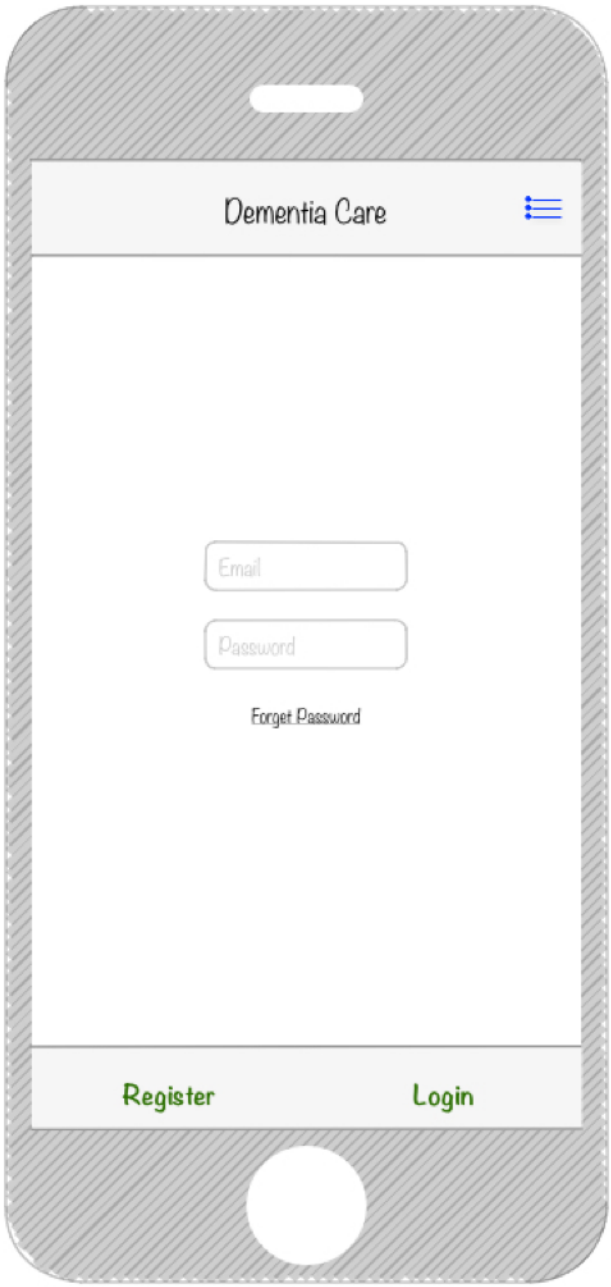

Figure 23. Login Interface

The login view allows user to login via email and password. It also allows user to jump to the register view to register an account. Moreover, there is a 'Forget Password' link to retrieve his/her password by answering several secret questions.

- **Homepage of Dementia Patients**

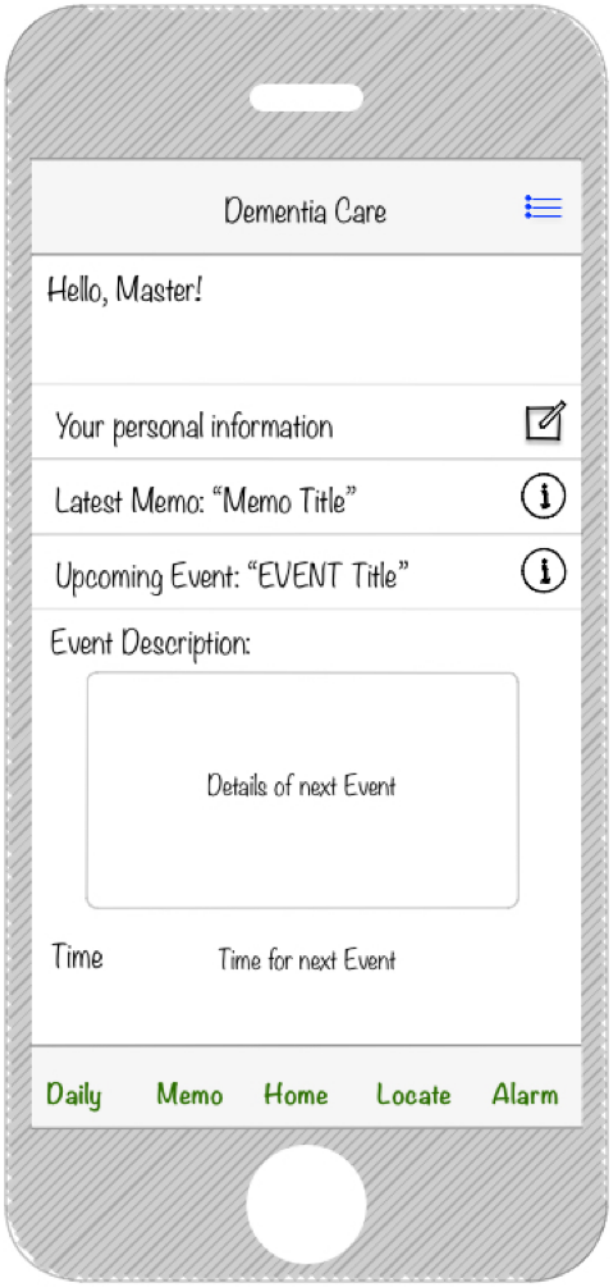

Figure 24. Homepage of Dementia Patients

This patient home view shows basic information of the patient, such as user name and details of next event. As the figure shows, the navigation design style is the traditional 'Tab bar menus' [8]. Although it is fixed on the screen, which means it occupies some screen space all the time, it still has some brilliant advantages. Firstly, compared with Swipe Menus, it is clearer and more direct. Secondly, the interaction is very simple, i.e. user can instantly jump to another page with a simple tap.

- **Daily Event View**

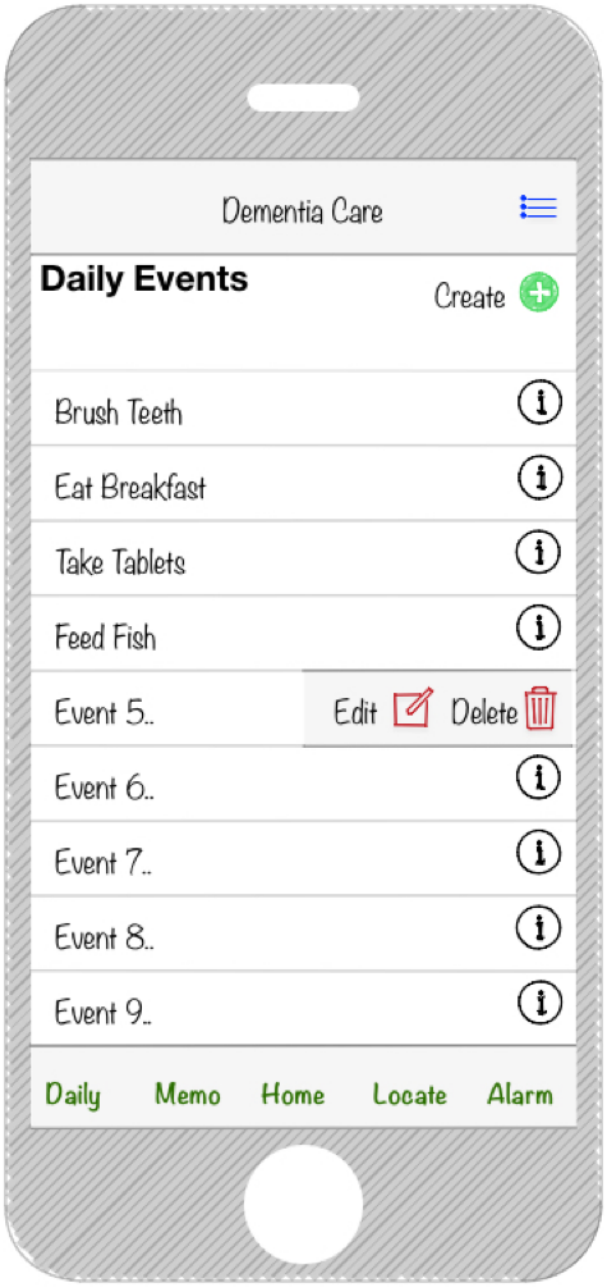

Figure 25. Daily Events View

Dementia patient user can jump into the Daily Events page by tapping the 'Daily' button on the navigation bar. This page shows all the daily routine events of this patient. In order to check the details of each event, one can press the info button. Daily event could be created by clicking the 'Create' button. By swiping left, the user can edit or delete this event.

- **Create Daily Event View**

Figure 26. Create Daily Event View

As shown in the previous page, user can enter Create Daily Event view by clicking the 'Create' button at the top right corner. In this view, user could set up daily routine reminders by entering details of an event according to the instructions.

- **Daily Event Reminding View**

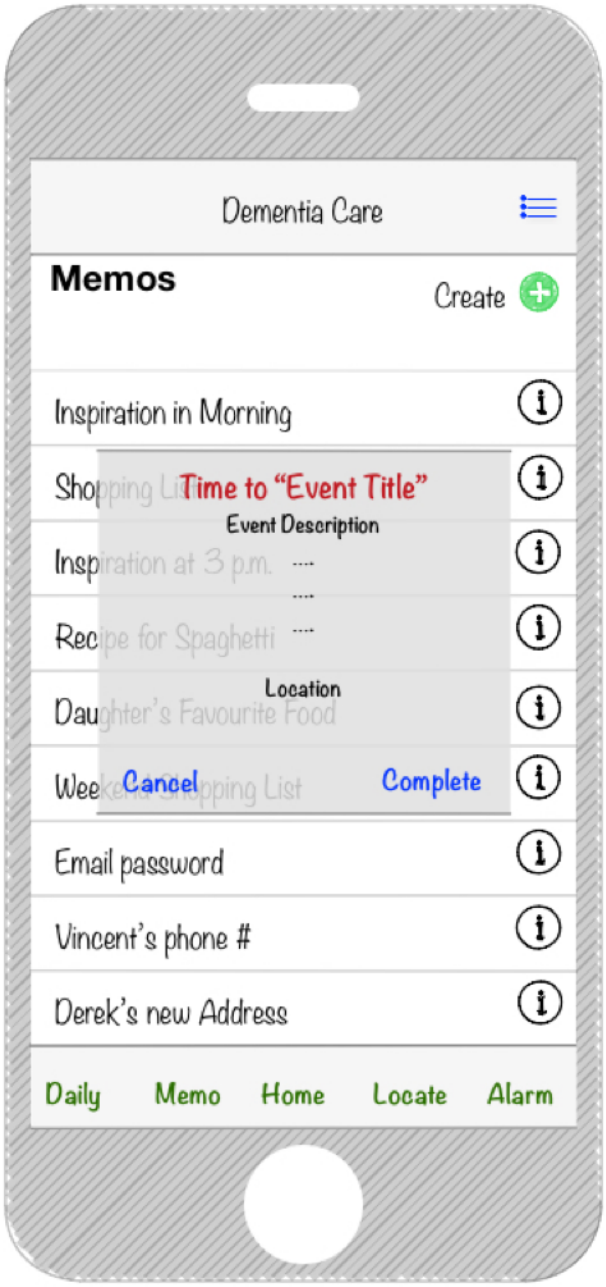

Figure 27. Daily Event Reminding View

If it is time to do something, there will be a popup window to remind the user. The patient should press 'Complete' when the thing is done and 'Cancel' when he/she dose not want to do it at this time.

- **Pull-down Menu View**

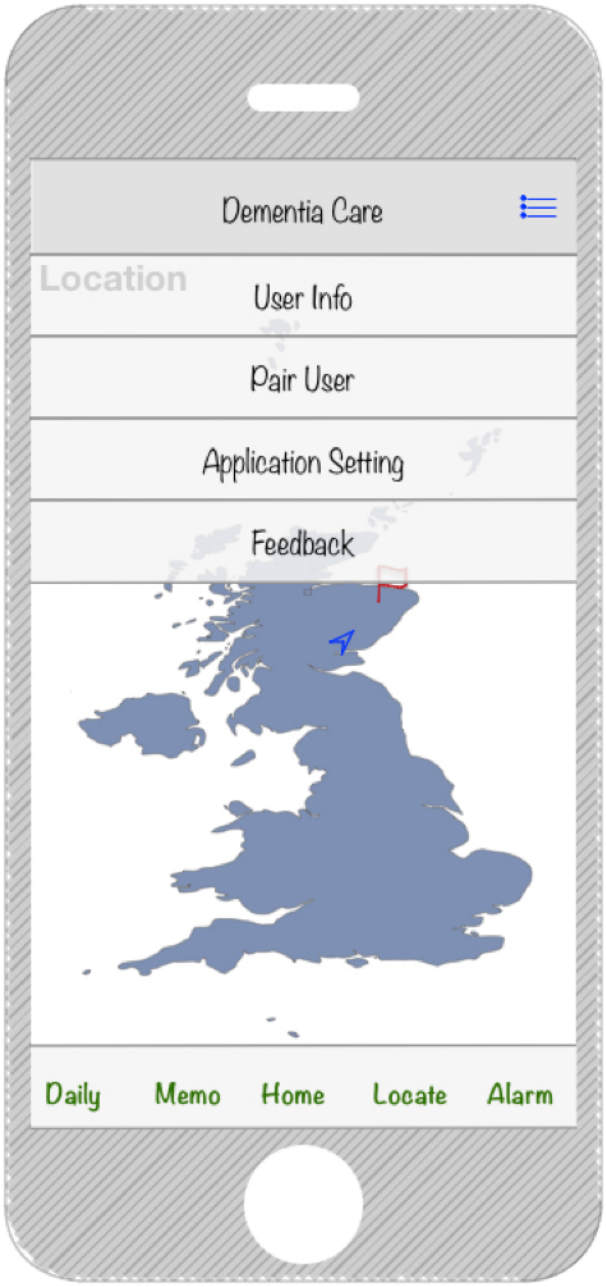

Figure 28. Pull-down Menu View

As shown, if user click the blue ‘Menu’ button at the upper-right corner, there will be a pull-down menu. In this menu, user can check and edit all his/her user information, and pair other corresponding users. User also can enter the application setting and write feedback by using this pull-down menu.

- **Emergency Button View**

Figure 29. Emergency alarming View

Dementia patient can enter this view by clicking the 'Alarm' button on the bottom navigation bar. By clicking the red button, the application will send emergency alert to all the patient's caregivers. If the emergency alarm is triggered by mistake, the user would be able to send a Cancel message to all his/her caregivers. In this design, patient can send a help request in a simple tap in emergency cases while mistakenly triggered situations are considered too.

- **Homepage of Caregiver**

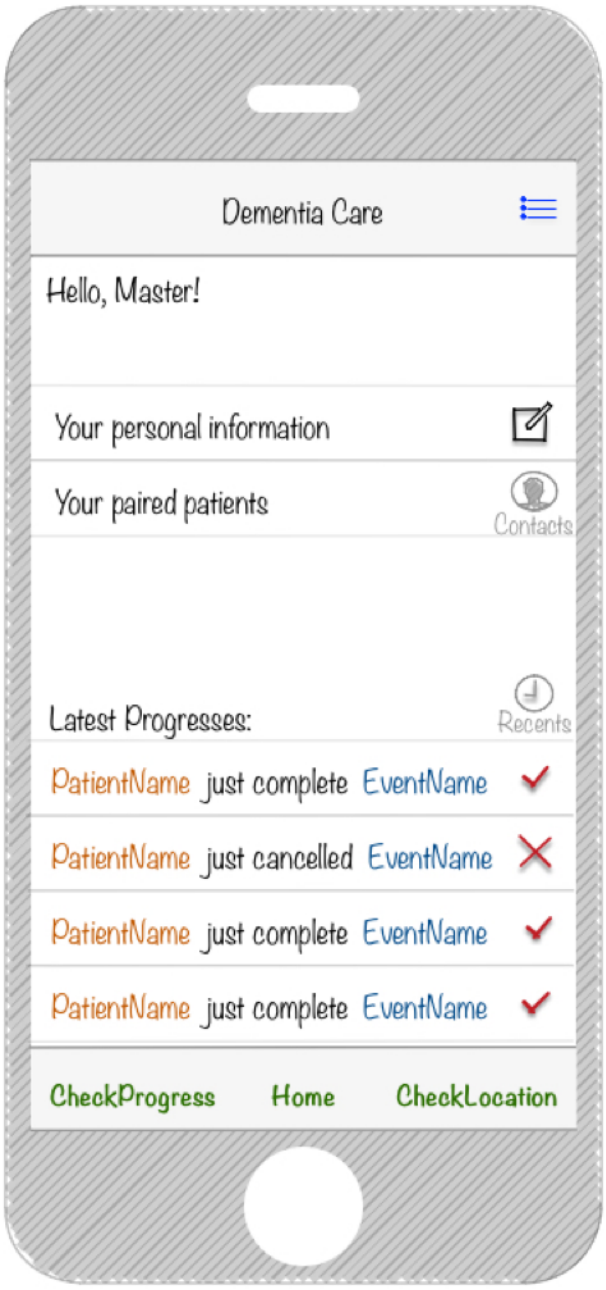

Figure 30. Homepage of Caregiver

When a caregiver logged in via the login page, he/she will be brought to this homepage. In this page, the caregiver can see his/her name on the screen, and check his/her personal information as well as associated dementia patients. The latest progresses of daily events of all the patients of the caregiver will also be displayed on the home page.

- **Pair Interfacer**

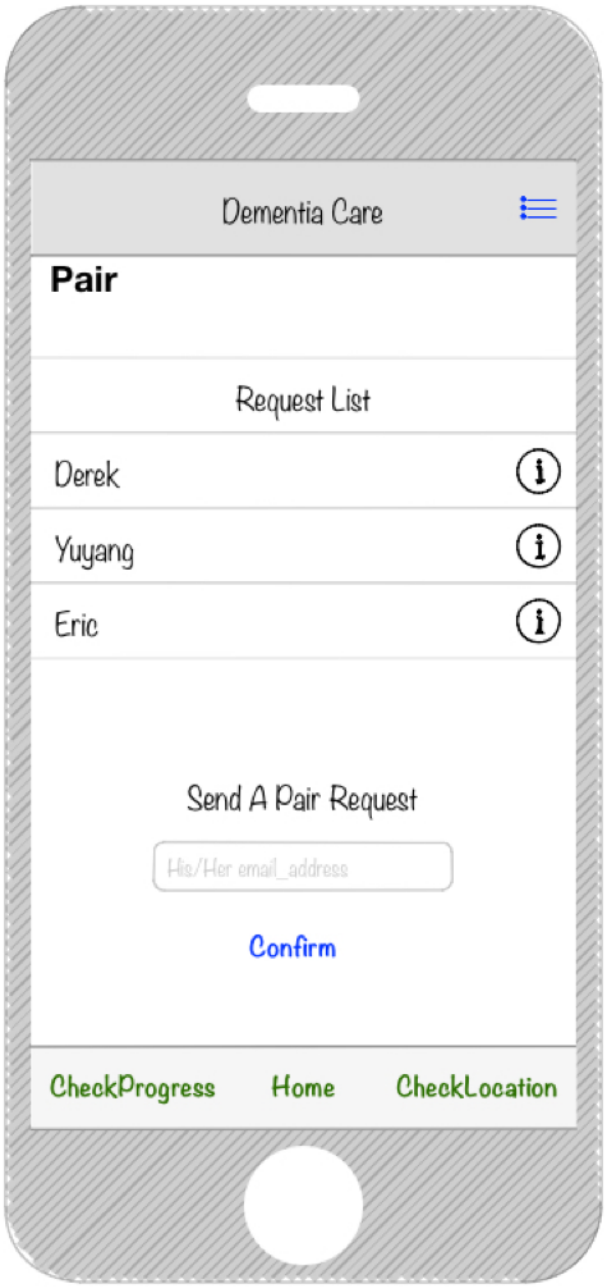

Figure 31. Pair Interface

The pair interface not only shows the request list, but also allows user to send a pair request. By clicking the info button of each item in the request list, user can see details of the request and then decide whether to pair with the applicant or not. Also, user can send a pair request to another user (different user type) by entering his/her email address.

4.6. Evaluation Design

4.6.1. Evaluation Criteria & Assess Methods

- Specification Completion & Validation

  The threshold of the evaluation is to check whether the final product follows the specification. Because the product will be desired only if it satisfies the specification. Thus, the product will be reviewed against its specification.

- Number of features completed

  The completion of anticipated features will also be evaluated as a criterion. All the fulfillments and absences of expected features will be summarised in a detailed analysis.

- Efficiency

  Generally, the user experience of an efficient program is often enjoyable. Efficiency is also indispensable to this project. It will be examined by checking whether a function performs with the minimum use of whatever resource that is necessary for it, function by function [9].

- User friendliness

  The human-computer interaction is vital to a product. Because every subtle improvement in human-machine interaction could give a product big advantages in competition and increases stickiness with its user base. Hence, the user friendliness is also an important criterion in this evaluation. It will be investigated by interviewing supervisor and making comparison with existing similar products.

- User's feedback

  User's feedback is valuable for product bug discovery and future improvement. It will be considered to evaluate whether the system is successful. The assess method is handing out a small number of questionnaires.

- Supervisor's feedback

  This criterion is also crucial to the product's evaluation, since supervisor could provide the project with very useful advice. Unlike user's feedback, the professional advice could help the product's future improvement in another perspective.

4.6.2. Evaluation Participants

Sebastian Coope, Michele Zito and I will be involved in the evaluation. A number of classmates might be invited to fill a questionnaire to provide their user feedback.

4.6.3. Testing Plan

The testing sets will be written before the programming phase, and then be used as a beacon of coding and refactoring. The test sets will be composed of integration testing, unit testing. The integration testing suite will be used to ensure all the code cooperates. It always helps with localisation of differences between expectation and reality. The unit testing suites will verify that every unit of code behaves as expected. This test set might run multiple times a day, in order to achieve a high-quality system and a clean architecture. Moreover, testing will be divided into functional tests and non-functional tests. The functional tests will be specific yes or no tests based on functional specification, while the non-functional tests value the quality of performance. The non functional tests will be composed of stress test, usability test and security test.

4.6.4. Expected Conclusion

Firstly, all the functions will be performed successfully. Secondly, the user-interface will be simple and clear to most users. Thirdly, the efficiency of the system will be acceptable.

4.7. Ethical Use of Data

4.7.1. Data Usage

The data will be required in this project are simulated data, such as daily events and emergency events. It will follow the University Policy on ethical use of human data and human participants.

4.7.2. Human Participant

I will use myself as a testing participant to test different functions of 'Dementia Care', such as locating and reminding. The project will exactly follow the University Policy on ethical use of human data and human participants.

4.7.3. Ethical Issue In Future

Currently, the data usage and human participants are used as above mentioned. However, if the application is put into markets, there will be real data and other human participants involved. The data protection will be considered. All the data transmission will be encrypted. The protection of personal information will be the first priority.

5. Realisation

In this section I will cover how the project was implemented from the design into a functional application. However, I will only cover the key aspects of this system, since the final realisation consisting of over fifty Objective-C files, a dozen PHP files and thousands lines of code. These will be explained with the aid of code snippets and screenshots where required. The full source code can be viewed within the zipped archive, which were submitted with this report.

5.1. Development Environment Setup

During the project implementation, the first thing I needed to do was setup a development environment which could provide all the tools the system required. Firstly, I created a subdomain name '*DementiaCare*' under my personal domain 'c*loudcampus.xyz*', which I used last year for the group project. Then, I also registered a new hosting server for the data storing. In the meantime, I also configured the FileZilla, which enables me to transfer files between local system and remote host. After these, the server is ready.

Alongside of the server preparation, I also decided to join the Apple Developer Program membership to enable myself to use the APNs service for the later emergency alarming system.

5.2. Database Construction

The remote database is a MySQL database. The database as well as tables creation were completed by SQL commands, which were written in several PHP files. Alongside controlling the database by codes, I also utilised PHPMyAdmin to manage the database. During the database creation, I designed the database on scratch paper first, which made the coding phase became much more easier since I only needed to translate handwritten design into SQL codes. Here is a snapshot of the structure of the database:

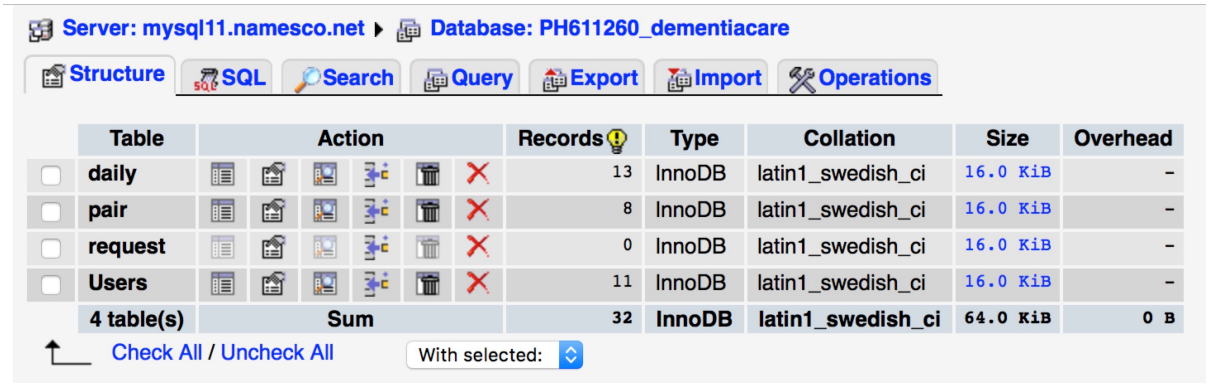

Figure 32. Remote Database Structure

There are only four tables, which are used to store daily routine events, linked relationships between patients and caregivers, link requests and user accounts.

5.3. User Interface Creation

The user interface of *Dementia Care* was created in storyboard of Xcode. The relationships of different scenes and transmitting segues are defined in the storyboard.

In the remaining pages of this section, I will introduce a number of key interfaces of this software, combing with screenshots of the running application.

- **Login Scene**

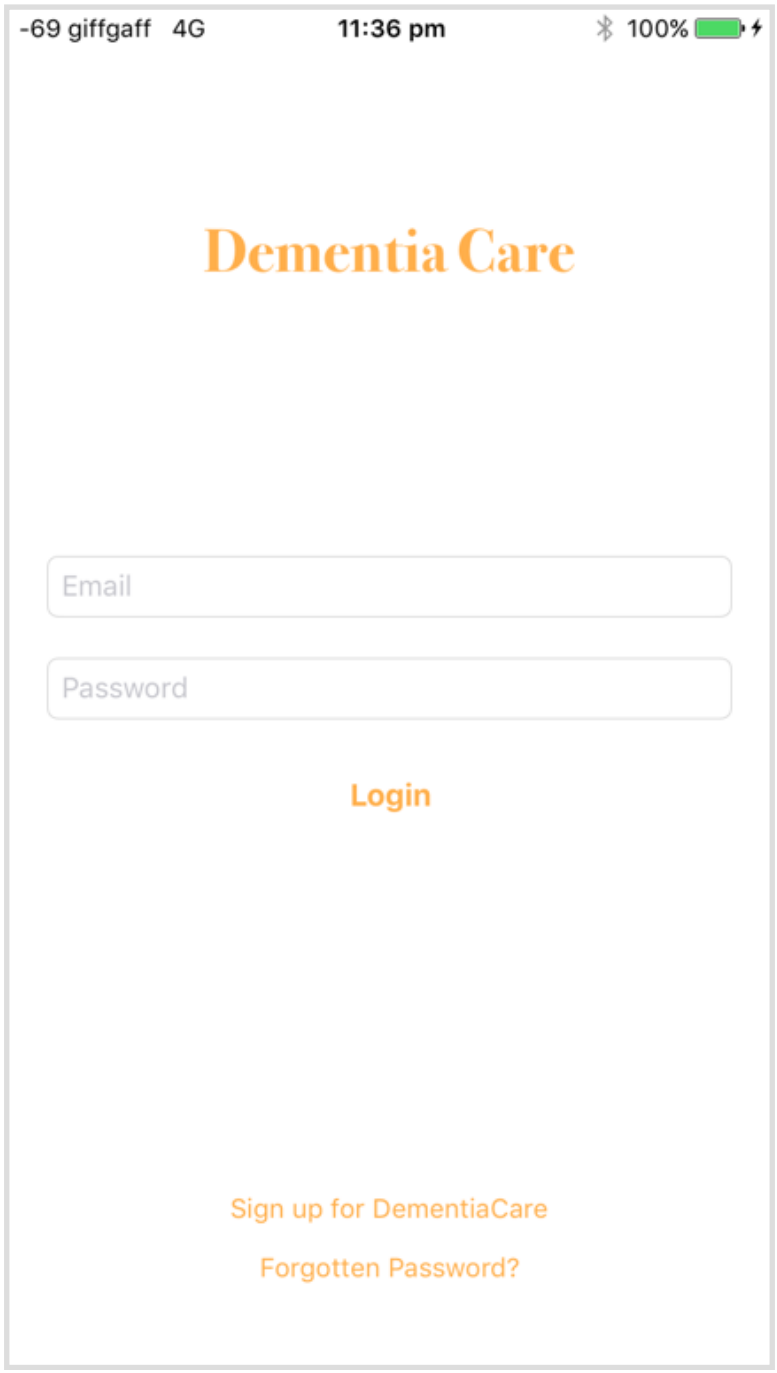

Figure 33. Login Scene

The final login view has two textfields to indicate users to input their login credential. User can also register a new account or rest his/her password with the bottom buttons. Once the user logged in, the device will remember his/her login state and user type, which enables the application to auto-login (skip this scene, directly navigate to his/her homepage) for the further usage. Rather, all the login states will be removed when signing out.

- **Register Scene**

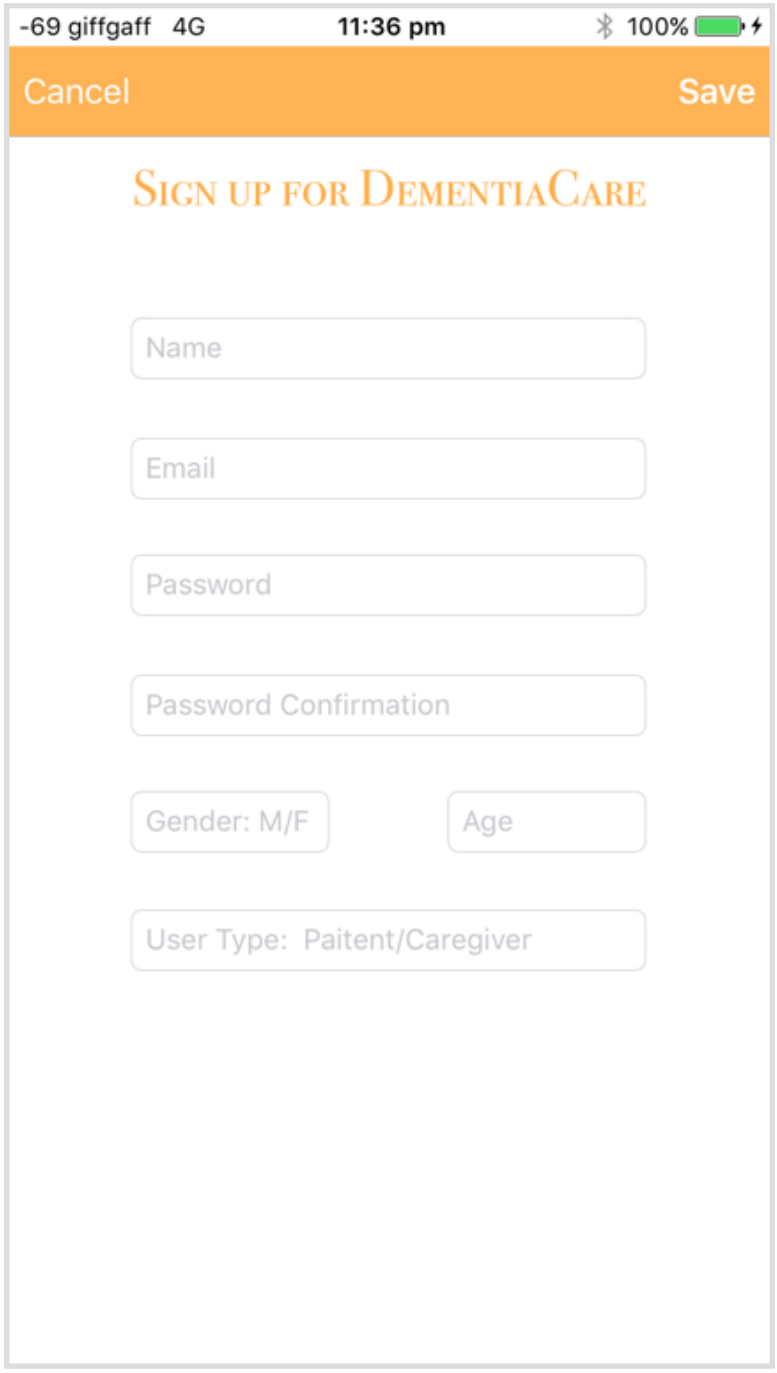

Figure 34. Register Scene

If one is new to the application, he/she will be able to sign up for *Dementia Care* by tapping the register button in login scene. In this page, the user should complete the form as indicated. Once the user clicked the *Save* button, the application will run a local grammar checking before submit it to the cloud. This design could not only reduce the server pressure, but also the times of data interchanges.

- **Homepage of Patients**

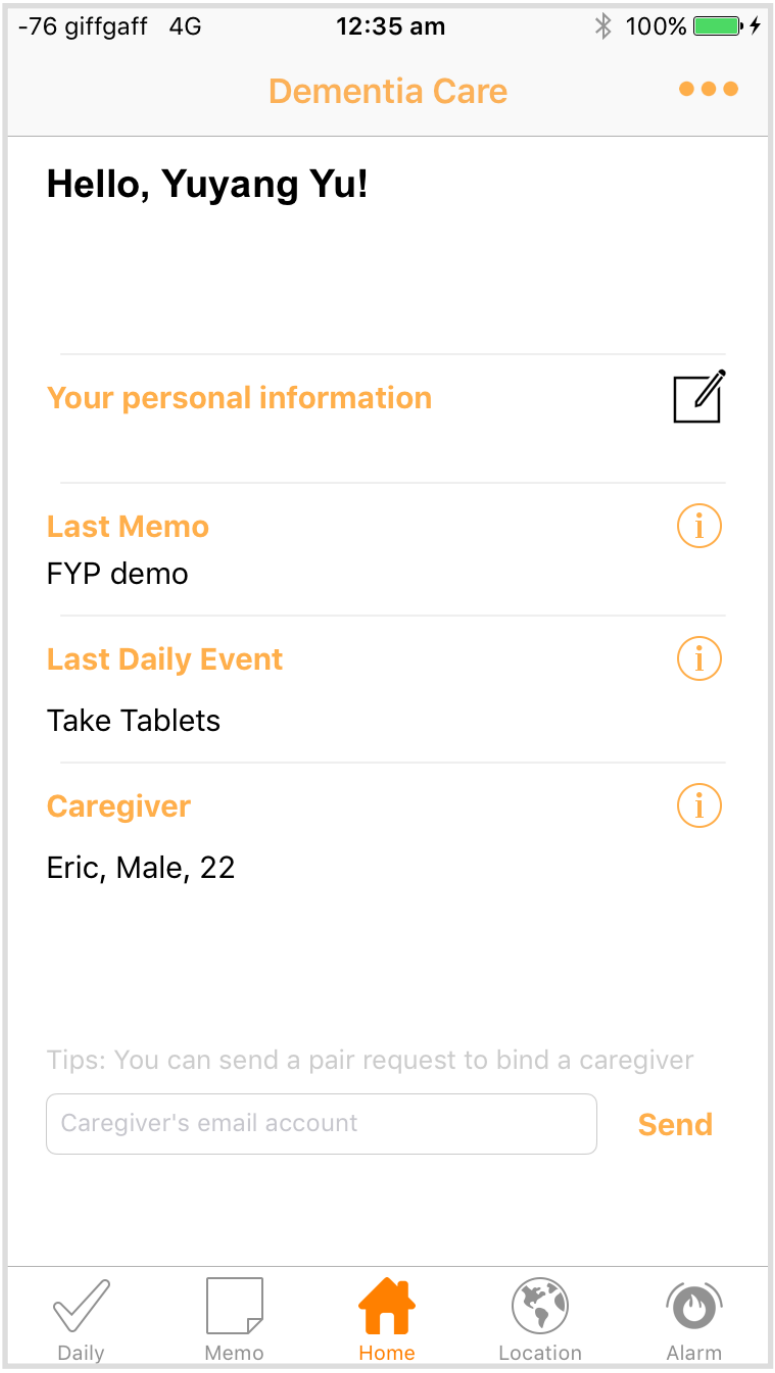

Figure 35. Dementia Patient Homepage

Once user logged in, the system will navigate him/her into the patient homepage if his/her user type is patient. As shown, the homepage will show several basic information of the user, including name, last memo, last event as well as his/her caregivers. User can send caregiver link request by using the bottom textfield and *send* button. User can also enter the application settings, application feedback and sign out by tapping the drop-down menu in the upper-right corner. Also, the interface is designed in tab bar style, user can enter other four major functions by using the bottom tab menu.

- **Homepage of Caregivers**

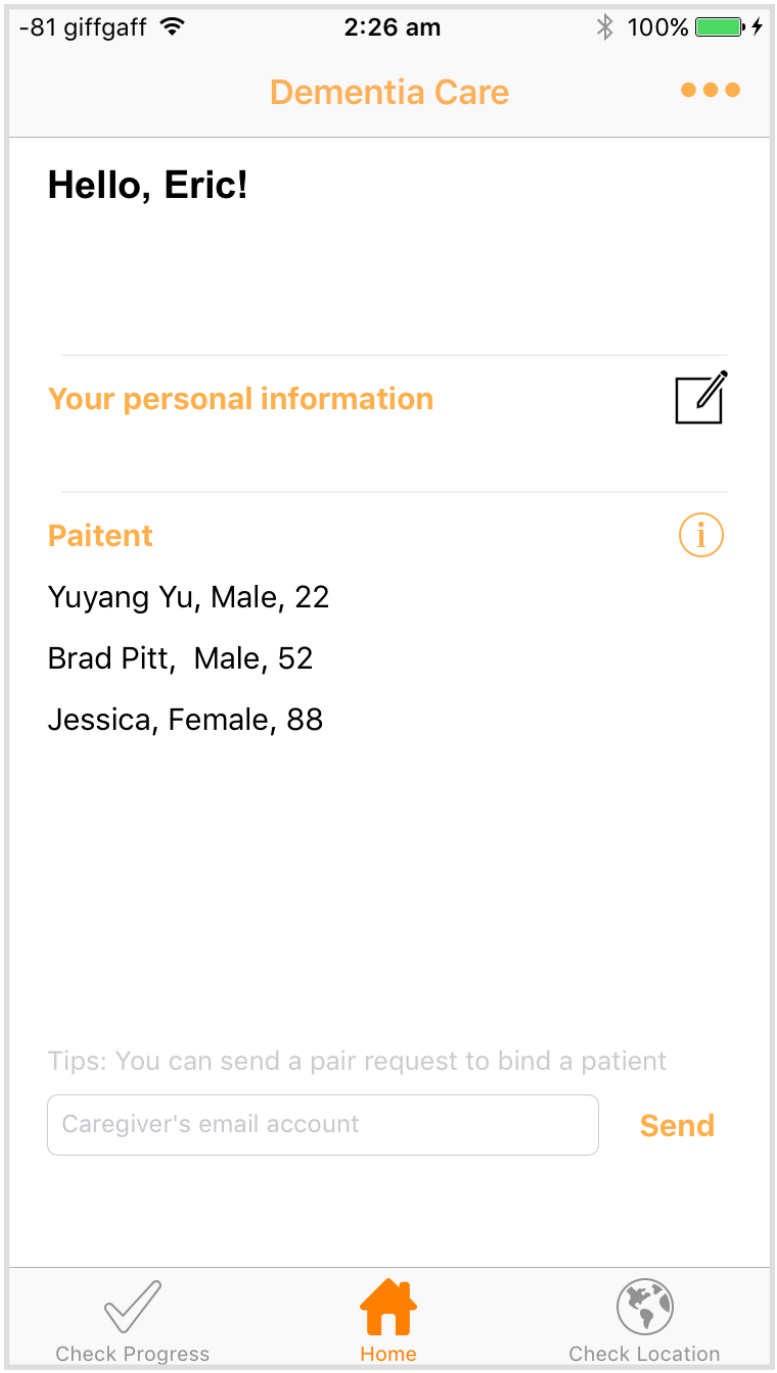

Figure 36. Homepage of Caregiver

When caregiver logged in the application, he/she would be taken into this caregiver's homepage. This interface is very similar to patient's homepage, this design is intended to interface consistency. As shown in the picture, there also are some basic information of the caregiver. Caregivers can also send patient link requests to patients in this scene. The patient and request management view can be entered by click the detail button in Patient zone. Plus, there are only three tab in the tab bar menu, which enables caregiver to check his/her progresses as well as locations and back to the home page.

- **Daily Routine Scene**

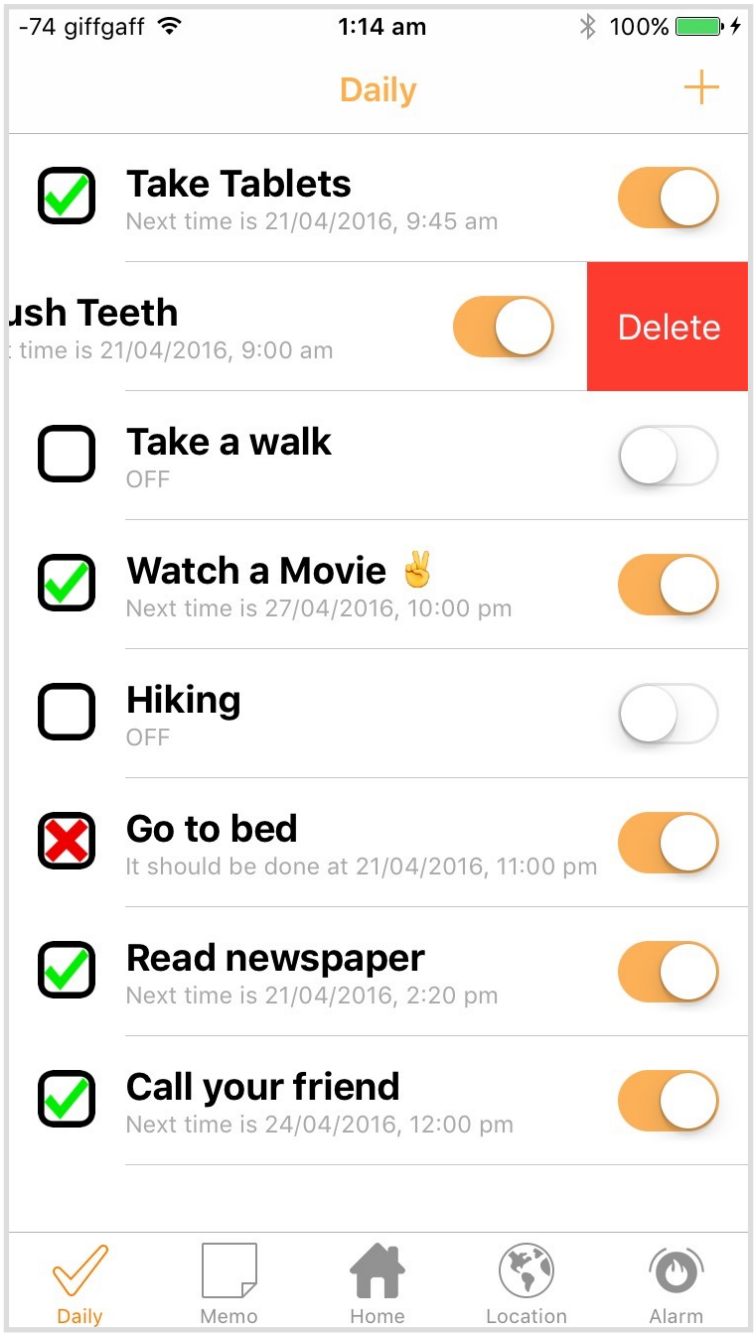

Figure 37. Daily Routine Scene

Back to the interfaces of patients, the first major section is daily routine. User can enter this section by tapping the *Daily* button on the tab bar. As shown, there are a number of daily routine events. The left side checkboxes has three status: check mark indicates completed, cross mark indicates waiting for completion and blank checkbox indicates the reminder has been turned off. The right side switcher enables user to turn ON/OFF each events. Moreover, user can delete an event by swiping left and then clicking *Delete*. User can also check one event detail and then update completion by clicking that event cell. The upper-right *add* button enables user to set more event reminders.

- **Daily Routine Creation Scene**

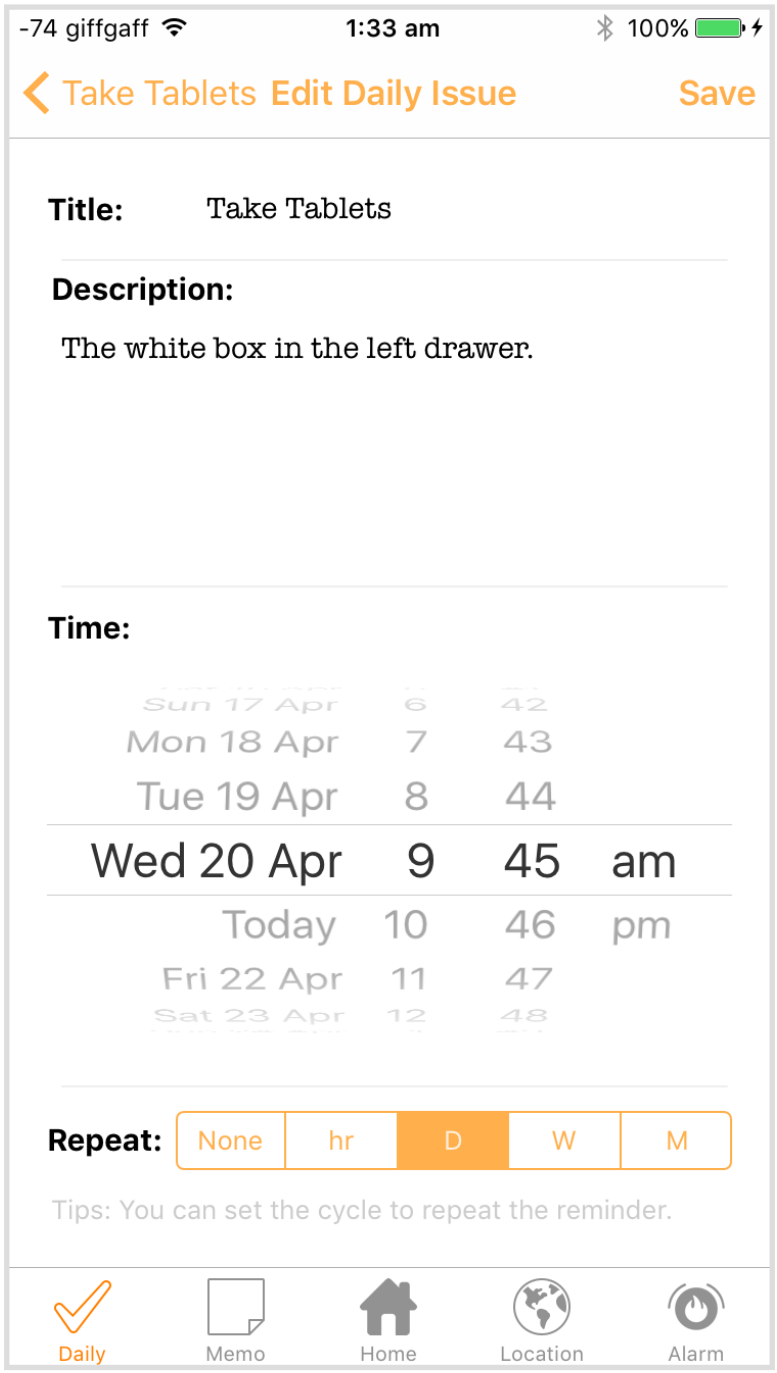

Figure 38. Daily Routine Creation Scene

As above mentioned, patient can enter the daily routine creation scene by tapping the *Add* button. There are title textfield, description textview, time picker as well as a repeat segment scheduler. These kits work together to enable user set up a daily routine reminder as they wanted.

- **Daily Routine Detail Scene**

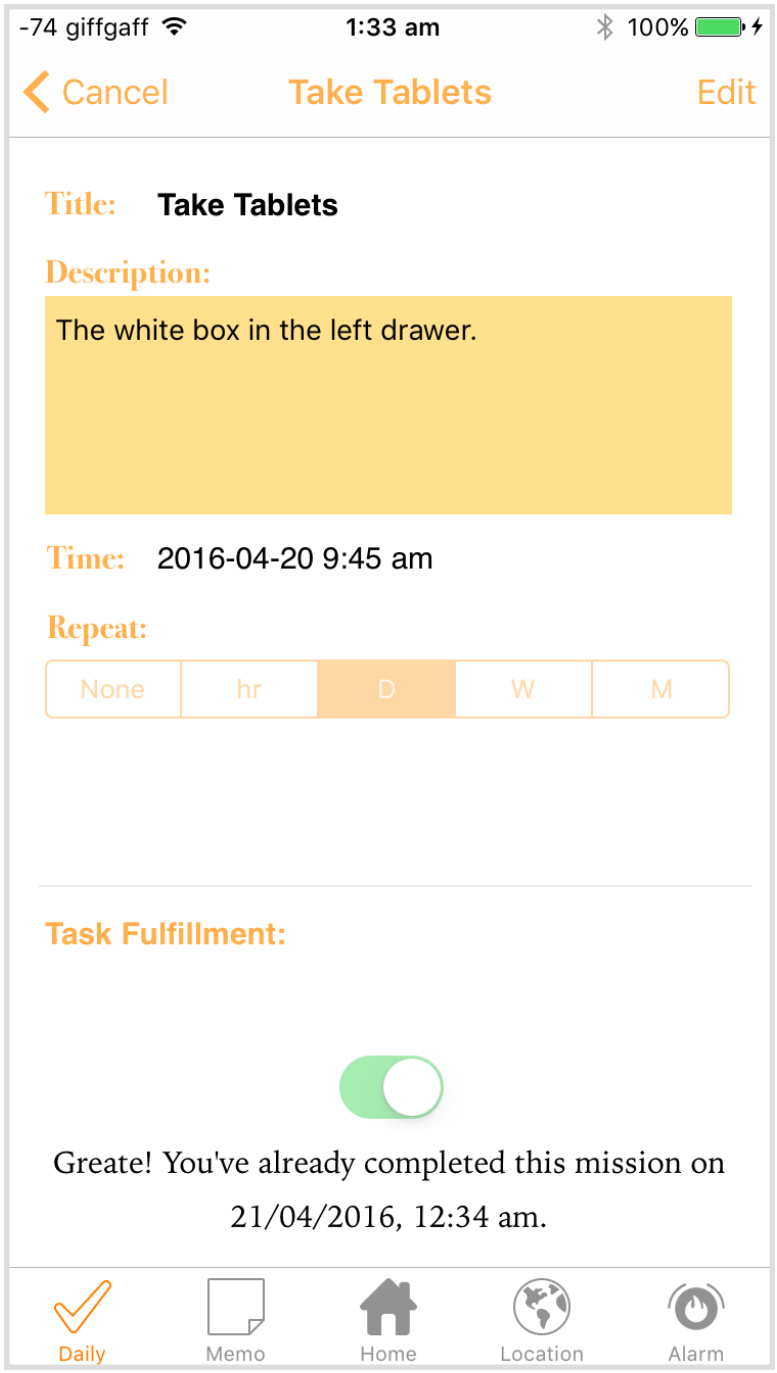

Figure 39. Daily Routine Detail Scene

Also, patient can view every daily routine's details and update his/her completion by entering the detail scene. There are title, description, original set time and repeat cycle. Once the task is time to be done, the patient would be able to switch the switcher to indicate completion. Once the task is completed, the switcher will be disabled until the task is ready for completion again.

- **Daily Routine Reminding View**

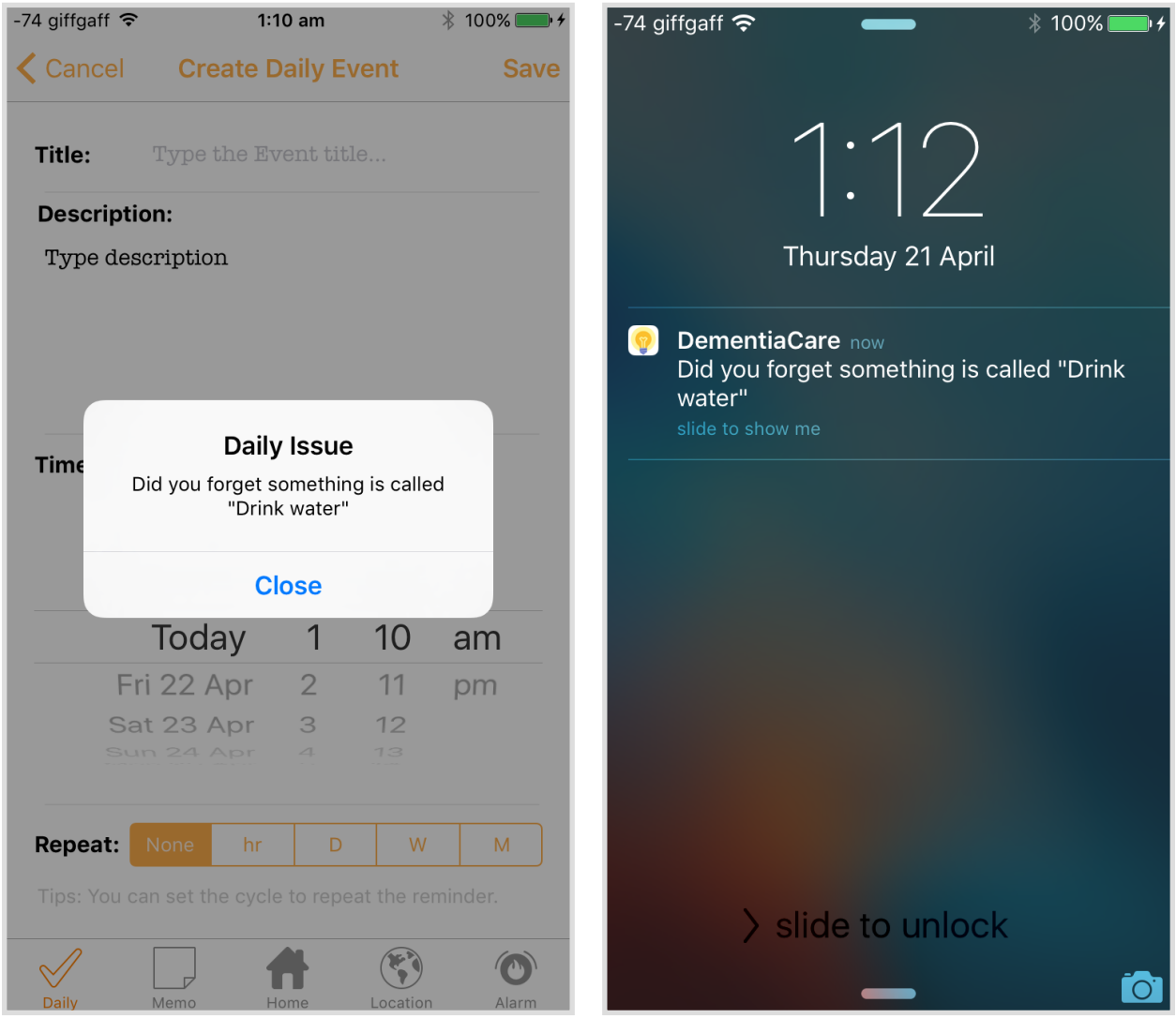

Figure 40. When reminder is triggered

Above two picture shows the effects of reminders. When it is time to do something, the scheduled local notification will be triggered. If the user is using the application, there will be a popup window to remind him/her with the title of the event. Otherwise, *Dementia Care* will push a normal notification to the user to remind the user that it is time to do something.

- **Memo and Memo Creation Scene**

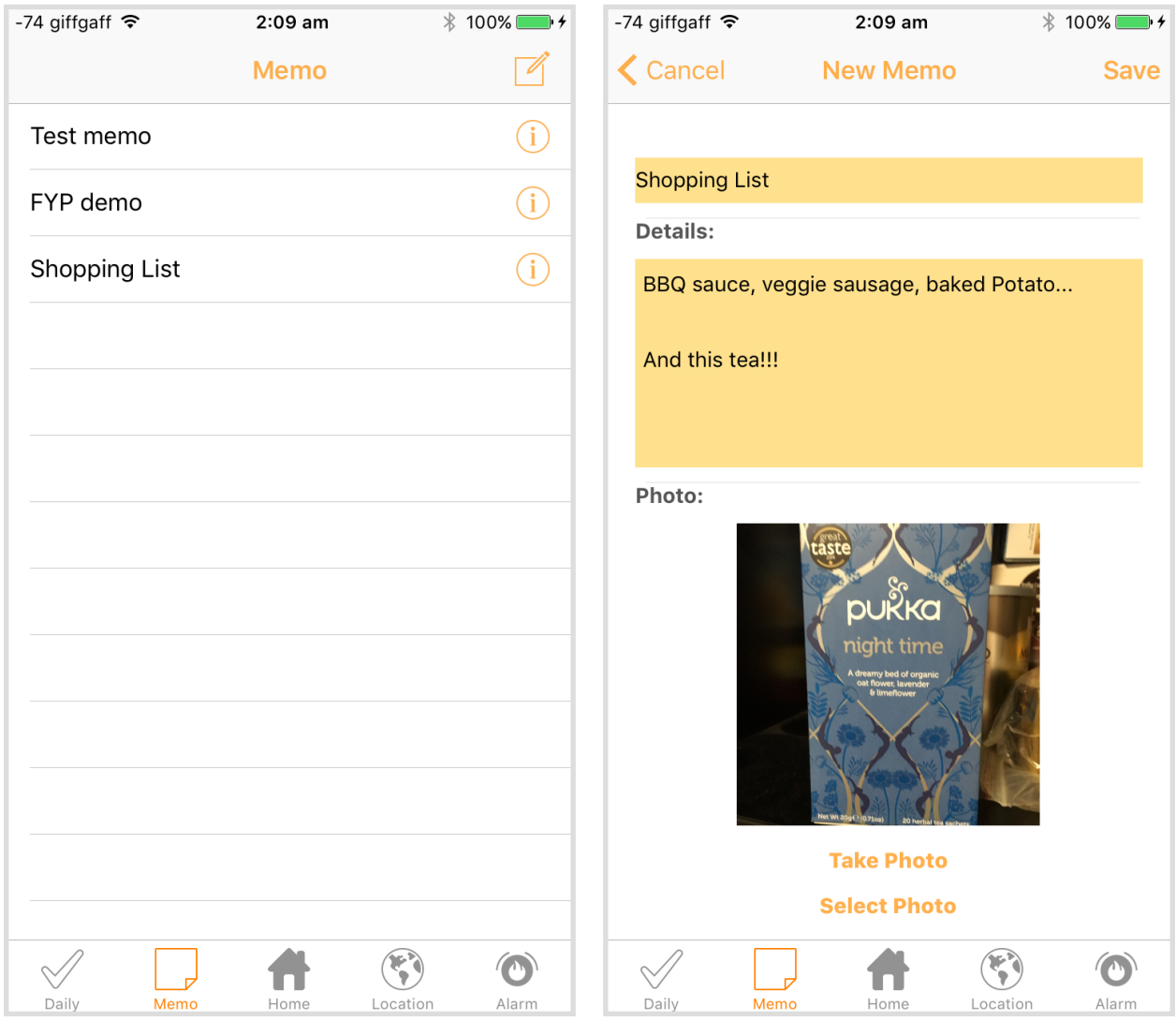

Figure 41. Memo and Memo Creation Scene

The second major function for patients is memorandum. User can enter the Memo scene by tapping the *Memo* button at the tab bar. The Memo scene contains a list of all the memos created by the dementia patient. He/she can then check the details of each memo by tapping the detail button of each row on the right side.

Dementia patients are also enabled to enter the Memo Creation scene by tapping the *Compose* button on the top right corner. In this interface, there are textfield for title and textview for details. Users can also using device camera to take a photo or read one from the system camera roll by clicking corresponding buttons.

- **Location scene and Address Book Insertion**

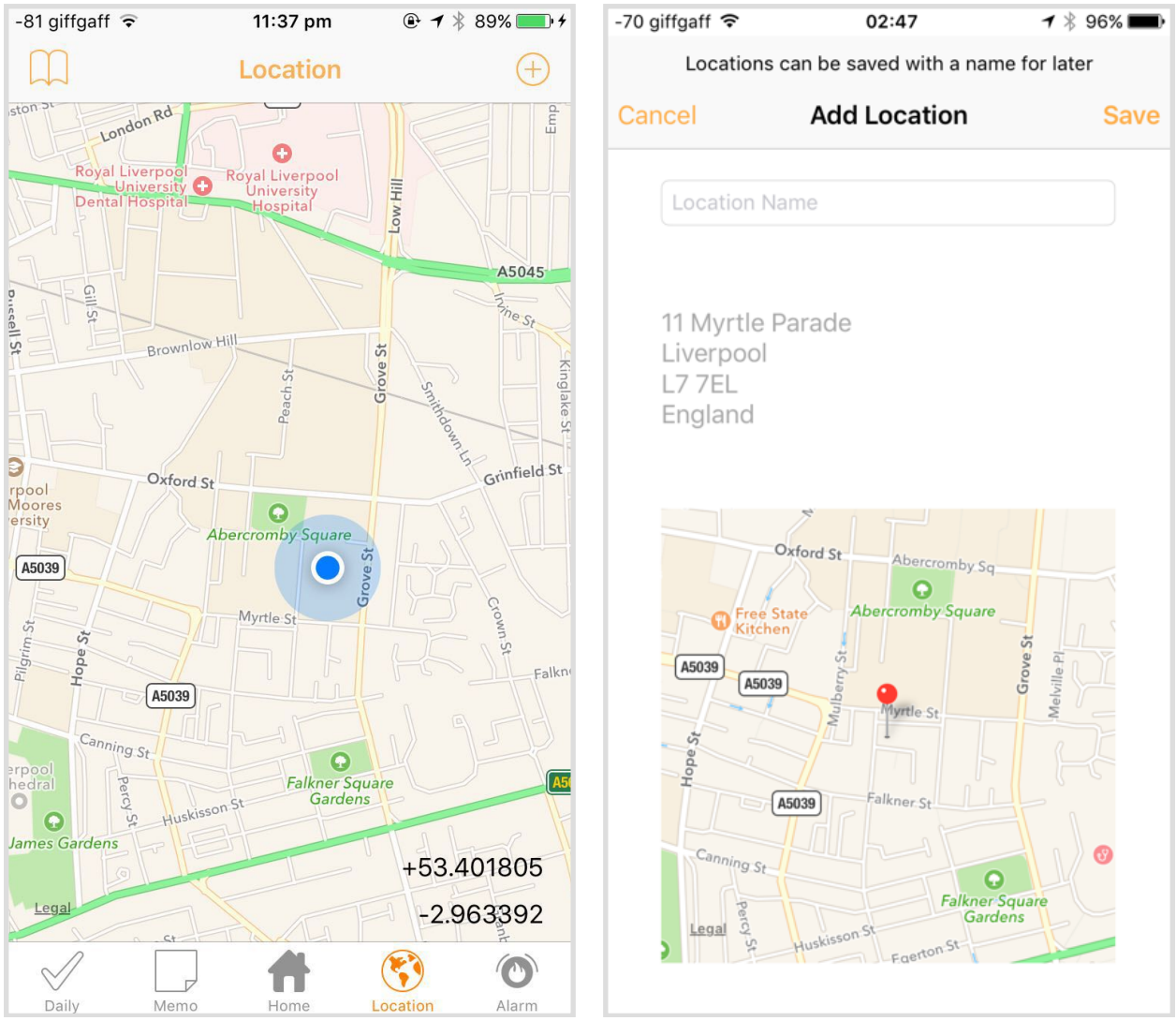

Figure 42. Location Scene and Address Book Insertion

The third section designed for dementia patients is self-locating map, together with and address book. In the Location view, the interface consisted of a map view, an address book button, an insertion button and longitude label as well as latitude label.

By clicking location inserting button, user will be navigated to the address book insertion view. In this interface, the textfield enables user to name the address, the textview will show the geo-location details of the user current location.

- **Emergency Scene**

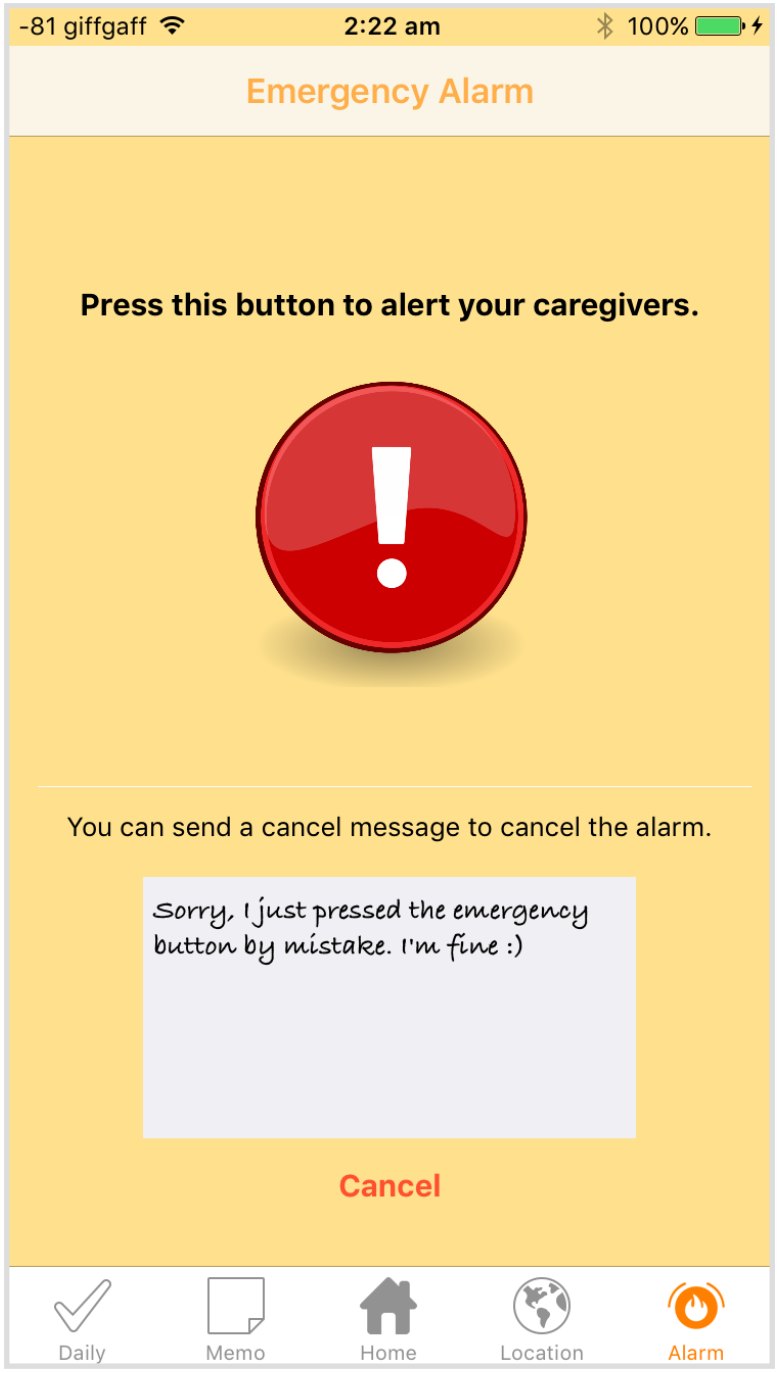

Figure 43. Emergency Scene

The last primary function is emergency alarming system. As shown in the figure, patient can enter this view by tapping the *Alarm* button in the tab menu. There is a striking emergency button, which is realised by a customised UIButton. There is also a textview for editing cancellation message and a *Cancel* button for sending cancellation messages.

- **Locating Patients Scene**

Figure 44. Check Location Scene

Finally, take a look at the interface for locating patients. This is designed for caregivers, it is consists of a map view, together with several annotations which indicate the real-time locations of patients of the caregiver. By tapping the annotation, caregiver can view the patient's name and location name.

5.4. Login System Realisation

The first part of the software is the login system. Both patient user and caregiver user will enter the login scene whenever they launch the application. The login interface will ask users for their user accounts as well as their passwords. The user account should be an email address, and the grammar checking will be done locally in order to reduce server pressure and save time if typo occurs. To verify the user's credential, I created a PHP page to receive posted credential and check with *User* table in database. The verification process is very simple: PHP will select entries from the table where account matches the posted account and password matches the posted password. If the query has result, PHP will return the account information back to the application and indicate login succeed. Otherwise, PHP will return a message to indicate login failure.

```
$sql = "SELECT Email FROM Users WHERE Email = '$_POST[email]' and Password = '$_POST[password]'";
$result = mysql_query($sql, $con);
$num = mysql_num_rows($result);
// Login succeed! Return account information
if($num) {
  $sql = "SELECT Name, Email, Gender, Age, user_type FROM Users WHERE Email = '$_POST[email]'";
  $result = mysql_query($sql, $con);
  $t_result = mysql_fetch_array($result);
  echo '{"response":[{"success":"1","name":"';
     echo "$t_result[0]";
     echo '","email":"';
     echo "$t_result[1]";
     echo '","gender":"';
     echo "$t_result[2]";
     echo '","age":"';
     echo "$t_result[3]";
     echo '","type":"';
     echo "$t_result[4]";
     echo '"}]}';
}
else {    // Login Failed! Indicates there is no such user
  echo '{"response":[{"success":"0"}]}';
}
```

Figure 45. Account credential verification in PHP

Also, as shown in above snippet, the data interchange from PHP to Objective-C is in JSON format. JSON enables me to fetch and use data from the remote database.

Back to the client side, the login credential were uploaded by Post method. I almost used this method for all uploading work in this application. Here is a snippet of data interchange from Objective-C to PHP:

```
// Construct a string to hold post data, and then convert to NSData
NSString *post = [[NSString alloc] initWithFormat:@"email=%@&password=%@",[_email text], [_password text]];
NSData *postData = [post dataUsingEncoding:NSASCIIStringEncoding allowLossyConversion:YES];

// Get data length
NSString *postLength = [NSString stringWithFormat:@"%lu", (unsigned long)[postData length]];

// Locate the PHP file
NSURL *url = [NSURL URLWithString:@"http://www.cloudcampus.xyz/DementiaCare/login.php"];

// Construct the url request
NSMutableURLRequest *request = [[NSMutableURLRequest alloc] init];
[request setURL:url];
[request setHTTPMethod:@"POST"];
[request setValue:postLength forHTTPHeaderField:@"Content-Length"];
[request setValue:@"application/json" forHTTPHeaderField:@"Accept"];
[request setValue:@"application/x-www-form-urlencoded" forHTTPHeaderField:@"Content-Type"];
[request setHTTPBody:postData];

// Get returned data
NSError *error;
NSHTTPURLResponse *response = nil;
NSData *urlData = [NSURLConnection sendSynchronousRequest:request returningResponse:&response error:&error];

// Output response status code
NSLog(@"Response code: %ld", (long)[response statusCode]);
```

Figure 46. Objective-C posts data to remote PHP

Additionally, as already mentioned, the login view will appear whenever the user launches the application. However, in order to improve the user experience, the application will automatically dismiss this view and navigate the user to his/her homepage if the user already verified his/her account credential once. That is to say, users do not need to type their email accounts as well as passwords over and over again once them reopen the application. This function is achieved by storing user's account as well as user type in NSUserDefaults when the first time they successfully logged in. There is a code snippet:

```
if (success == 1) {
    NSLog(@"Login Success");
    // Set user state
    [[NSUserDefaults standardUserDefaults] setObject:email forKey:@"email"];
    // Patient
    if (type == 1) {
        [[NSUserDefaults standardUserDefaults] setObject:@"Patient" forKey:@"type"];
    }
    else {
        [[NSUserDefaults standardUserDefaults] setObject:@"Caregiver" forKey:@"type"];
    }
    [[NSUserDefaults standardUserDefaults] synchronize];

    // Popup alert view to inform user
    [self alertSuccess:@"Logged in Successfully" :@"Login Success!"];
}
```

Figure 47. Auto-login

As shown, user state can be remembered when the user credential is verified. Thus, when the login view appears, application can directly navigate user to his/her homepage if there is remembered states. Certainly, the user state will be removed, if the user chooses to log out.

5.5. Daily Routine Reminding System Realisation

- **Schedule a reminder**

As described in interface realisation, the daily routine creation scene enables user to construct a reminder as wanted. The parameters are title, description, time and repeat cycle. Then we can schedule an *UILocalNotification* according to these parameters.

```
// Asign unique notification ID, which can be used to delete specified notif, etc
NSString *notifID = [[NSProcessInfo processInfo] globallyUniqueString];

Class cls = NSClassFromString(@"UILocalNotification");
if (cls != nil) {
    UILocalNotification *notif = [[cls alloc] init];
    // Set fire date
    notif.fireDate = [datePicker date];
    notif.timeZone = [NSTimeZone systemTimeZone];

    // Set alert
    NSString *alertBody = @"Did you forget something is called \"";
    alertBody = [[alertBody stringByAppendingString:[[self txtTitle] text]]
                                                stringByAppendingString:@"\""];
    notif.alertBody = alertBody;
    notif.alertAction = @"Show me";
    notif.soundName = UILocalNotificationDefaultSoundName;
    notif.applicationIconBadgeNumber = 1;
    [notif setUserInfo:[NSDictionary dictionaryWithObject:notifID forKey:@"notifID"]];
    NSLog(@"Set notifID%@", notifID);

    // Set repeat cycle
    NSInteger index = [scheduleControl selectedSegmentIndex];
    switch (index) {
        case 1:
            notif.repeatInterval = NSCalendarUnitHour;
            cycle = 1;
            break;
        case 2:
            notif.repeatInterval = NSCalendarUnitDay;
            cycle = 2;
            break;
        case 3:
            notif.repeatInterval = NSCalendarUnitWeekOfYear;
            cycle = 3;
            break;
        case 4:
            notif.repeatInterval = NSCalendarUnitMonth;
            cycle = 4;
            break;
        default:
            notif.repeatInterval = 0;
            cycle = 0;
            break;
    }
    [[UIApplication sharedApplication] scheduleLocalNotification:notif];
}
```

Figure 48. Schedule daily routine reminder

- **Daily Routine Storage and Consistency**

After scheduling reminders for the events, I needed to consider how to store them. I decided to create a *Daily* table for these events in local SQLite database. In this table, I stored their title, description, set time, repeat cycle, completion time, ON/OFF state and notification ID. These data enables me to further realise functions the project required.

Also, I created a *Daily* table in the remote MySQL to store every daily routines on the cloud. This enables caregivers to check their patients' daily routine progress. However, I knew it will waste some space to keep the almost same data in two databases. The reason why I employed a local SQLite database is I wanted to make this application become more practical to dementia patients. With a local database, the application can still work normally like a powerful reminder application or memo application, even if there is no Internet access. For example, a dementia patient needs to take a white pill every morning, it might be terrible if the reminding function is totally depended on Internet. Thus, I chose to sacrifices a little bit simplicity and space to make the *Dementia Care* more reliable.

In order to keep data consistency, all the local daily events will be uploaded to cloud for synchronisation when the daily event table is appeared or daily event is modified or deleted. These processes will be done automatically in the background. All the data uploading were done by PHP $_POST function, which is very similar to the code in Figure 46. Thus, if there is no internet, the updating will not influence the remote MySQL database, and the program will not stopped due to updating failure. The update can always be done at next time.

- **Determine Event's Completion**

The judgemental algorithm is vital to the daily routine reminder section, since it determines what to display to the patients as well as caregivers, whether the thing is done or not. Besides, this algorithm can also figure out the next deadline for repeating events.

In order to realise this function, I needed four arguments, which are the original setting time, repeat cycle, last completion time and current time. Thus, we can figure out the most recent fire time of this event by keeping adding up the original setting time according to the repeat cycle when the result time is before than current time. Then, by comparing the last deadline and last completion time, we can figure out whether the event is completed. Also, we can figure out the next fire time by revising the algorithm.

```
// last alert
- (NSDate *)recent:(NSDate *)time withDate:(NSDate *) ctime withCycle:(NSInteger)cycle {
    NSDate *temp = time;
    NSDate *current = [NSDate date];
    if (cycle == 1) {
        // temp time is not later than current time
        while ([[temp dateByAddingHours:1] compare:current] != NSOrderedDescending) {
            temp = [temp dateByAddingHours:1];
        }
    }
    else if (cycle == 2) {
        while ([[temp dateByAddingDays:1] compare:current] != NSOrderedDescending) {
            temp = [temp dateByAddingDays:1];
        }
    }
    else if (cycle == 3) {
        while ([[temp dateByAddingWeeks:1] compare:current] != NSOrderedDescending) {
            temp = [temp dateByAddingWeeks:1];
        }
    }
    else if (cycle == 4) {
        while ([[temp dateByAddingMonths:1] compare:current] != NSOrderedDescending) {
            temp = [temp dateByAddingMonths:1];
        }
    }
    return temp;
}
```

Figure 49. Figure out the last deadline

As shown, there are set time, repeat cycle, last completion time in the database, which enables caregivers to use similar methods to figure out their patients' daily progresses.

5.6. Memorandum System Realisation

Memorandum system allows dementia patient user create, check, edit and delete memos.

- **Picture storage**

These memo can contain a picture to achieve a better reminding effect. The picture will be stored in the application bundle, which means it can be fetched by its filename. Thus, I used the method *[NSProcessInfo processInfo] globallyUniqueString* to generate unique filenames, which can be stored in database in string format.

- **Memo storage**

Since there is no need to upload memo to the cloud, I directly stored them in the local SQLite database. The *Memo* table consists of Memo_ID, Memo_Title, Memo_Detail, Memo_Picture. The memo storing method is similar to the method used in daily event storage.

```
- (IBAction)saveMemo:(id)sender {
    // If the recordIDToEdit property has value other than -1,
    // then create an update query. Otherwise create an insert query.
    NSString *query;
    if (self.memoID == -1) {
        query = [NSString stringWithFormat:@"insert into memo values(null, '%@', '%@', '%@')",
                 self.txtTitle.text, self.txtDetail.text,
                 [[@"" stringByAppendingString:_imagePath] stringByAppendingString:@""]];
        // Execute the query.
        [self.dbManager executeQuery:query];

        // If the query was successfully executed then pop the view controller.
        if (self.dbManager.affectedRows != 0) {
            NSLog(@"Query was executed successfully. Affected rows = %d", self.dbManager.affectedRows);
            // Inform the delegate that the editing was finished.
            [self.delegate newMemoWasFinished];
            // Pop the view controller.
            [self.navigationController popViewControllerAnimated:YES];
        }
        else
            NSLog(@"Could not execute the query.");
    }
```

Figure 50. Saving memo to SQLite database

5.7. Locating System Realisation

Both two kinds of users can use a map to locate themselves. The map locating is realised by importing MapKit and CoreLocation frameworks in Xcode. The patient user can use a map and add locations to their own address books, while caregiver can check his/her patients' locations. There are several major issues:

- **Locating Self-location**

User can register a location manager to get his/her own location. For patient users, his/her location will be automatically updated to the remote MySQL database in the background. Here is code snippet for location updating:

```
// Report new location, i.e. the user has moved at least distanceFilter meters
- (void) locationManager:(CLLocationManager *)manager
     didUpdateToLocation:(CLLocation *)newLocation
            fromLocation:( CLLocation *)oldLocation {

    // Ensure that if we do something here, it is because we *are* in a different location
    if (([newLocation coordinate].latitude == [oldLocation coordinate].latitude) &&
        ([newLocation coordinate].longitude == [oldLocation coordinate].longitude))
        return;

    else {
        NSLog(@"MapVC new location: latitude %+.6f, longtitude %+.6f\n",
              [newLocation coordinate].latitude, [newLocation coordinate].longitude);
        [self showSpecifiedRegionAtLocation:newLocation];
    }

}
```

Figure 51. Update user current location

- **Inserting Annotations**

MKAnnotation protocol were used in this application to create labeled annotations, which is necessary for the locating system. For dementia patients, annotations should be inserted to the map view to represent addresses of their address book. For caregivers, annotation should also be used to represent their patients' real-time locations. Thus, I implemented the following method to insert annotations into map view:

```
LocationAnnotation *annotation = [[LocationAnnotation alloc]
                                  initWithCoordinate:coordinate
                                  title:name
                                  subtitle:revAddress];
[[self mapView] addAnnotation:annotation];
```

Figure 52. Map annotation insertion

5.8. Emergency Alarming System Realisation

As already mentioned, emergency system enables patients immediately send messages to their caregivers. Both emergency alerts and cancellation messages are implemented in the same way. There are several steps to achieve this function.

Firstly, the application will post the current user's account to the remote server.

```
NSString *post = [[NSString alloc] initWithFormat:@"user=%@",
                  [[NSUserDefaults standardUserDefaults] stringForKey:@"email"]];
NSURL *url = [NSURL URLWithString:@"http://www.cloudcampus.xyz/DementiaCare/apns.php"];
```

Figure 53. Posting User Account to Server

Then, the PHP page will get the $_POST variable (i.e., email account) to query the *Pair* table MySQL database. The database will return all the caregivers' device token, which enables the PHP page send APNs messages to these devices one by one. There is a code snippet of pushing cancellation messages to every caregivers:

```
$sql = "SELECT Users.token FROM pair INNER JOIN Users
            ON pair.Caregiver = Users.Email
            WHERE pair.Patient = '$_POST[user]'";
$result = mysql_query($sql, $con);
$num = mysql_num_rows($result);

// Iteratively send to all his/her caregivers
while ($x = mysql_fetch_array($result)) {
  $apns->connect(); // Connect
  $apns->addDT("$x[token]");  // deviceToken
  $apns->setText($_POST[cancel_msg]);  // Set Content
  $apns->setBadge(1);  // Set badge number
  $apns->setSound('keys.m4r');  // Set Ringtone
  $apns->setExpiry(3600);  // Set expriy time
  $apns->send();  // Send
  echo ' cancel message sent ok';
}
```

Figure 54. Pushing cancellation messages to all caregivers by APNs

Finally, when the application received a APNs message, *Dementia Care* will check whether the user is using it. If the user is not using *Dementia Care* at that moment, the application will push a notification as the APNs message defined. Otherwise, popup an alert view in the running application. Moreover, I also defined different ringtone for emergency alerts and cancellations. For example, every emergency push will trigger a 10 seconds loud and annoying alarming ringtone, which ensures the caregiver has the best chance of noticing emergency alerts in the first place. For more details about APNs handling method, here is the code snippet:

```
- (void)application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo {

    if (application.applicationState == UIApplicationStateActive) {
        // Nothing to do if applicationState is Inactive, the iOS already displayed an alert view.
        UIAlertView *alertView = [[UIAlertView alloc]
                                  initWithTitle:[NSString stringWithFormat:@"%@",
                                                 [[userInfo objectForKey:@"aps"] objectForKey:@"alert"]]
                                  message:@"" delegate:self cancelButtonTitle:@"OK" otherButtonTitles:nil];

        [alertView show];
    }

#if !TARGET_IPHONE_SIMULATOR

    NSLog(@"remote notification: %@",[userInfo description]);
    NSDictionary *apsInfo = [userInfo objectForKey:@"aps"];

    // Set alert
    NSString *alert = [apsInfo objectForKey:@"alert"];
    NSLog(@"Received Push Alert: %@", alert);

    // Set ringtone
    NSString *sound = [apsInfo objectForKey:@"sound"];
    NSLog(@"Received Push Sound: %@", sound);

    // Set badge
    NSString *badge = [apsInfo objectForKey:@"badge"];
    NSLog(@"Received Push Badge: %@", badge);

    application.applicationIconBadgeNumber = [[apsInfo objectForKey:@"badge"] integerValue];

#endif

}
```

Figure 55. APNs handler

5.9. Testing

Testing was an interesting predicament for this project since it was not clearly documentable. There is no written test cases for Objective-C. But I can still debug the application by using it and trying every functions in various ways.

Thanks for different data inputing, I found a huge bug in uploading data. The data will corrupt if the data is going to be uploaded to database and it contains quotation marks. This is because I did not run local checking for quotation mark and then managing to escape them.

The database and PHP pages were tested for correctness through means of attempting all possible ways of obeying and not obeying the rules of instructions. Except the bug I just mentioned, no eventful situation happened.

The daily routine completion determination algorithm had been tested with several sets of settings, including different start time and different repeat cycles. The algorithm is correct and robust.

Moreover, I tested the data consistency of synchronisation in abnormal situation. I cut off internet connection and made changes to daily routine events, the reminders can work according to the changes as usual. Then I connected the device into Internet again, when I reloaded the daily routine table, all the changes are updated into server again. The synchronisation success can be checked by using PHPMyAdmin.

6. Evaluation

As outlined in the design documentation, there are six evaluation criteria. In this section, I will evaluating the success of the project according to these criteria.

6.1. Criteria for success

- Specification Completion & Validation

The threshold of the evaluation is to check whether the final product follows the specification. Because the product will be desired only if it satisfies the specification. Thus, the product will be reviewed against its specification.

- Number of features completed

The completion of anticipated features will also be evaluated as a criterion. All the fulfillments and absences of expected features will be summarised in a detailed analysis.

- Efficiency

Generally, the user experience of an efficient program is often enjoyable. Efficiency is also indispensable to this project. It will be examined by checking whether a function performs with the minimum use of whatever resource that is necessary for it, function by function [9].

- User friendliness

The human-computer interaction is vital to a product. Because every subtle improvement in human-machine interaction could give a product big advantages in competition and increases stickiness with its user base. Hence, the user friendliness is also an important criterion in this evaluation. It will be investigated by interviewing supervisor and making comparison with existing similar products.

- User's feedback

User's feedback is valuable for product bug discovery and future improvement. It will be considered to evaluate whether the system is successful.

- Supervisor's feedback

This criterion is also crucial to the product's evaluation, since supervisor could provide the project with very useful advice. Unlike user's feedback, the professional advice could help the product's future improvement in another perspective.

6.2. Strengths of the Solution Produced

Firstly, I am going to evaluate the project in terms of specification completion as well as validation and number of features completed.

The anticipated software is an iOS App, which can run on iPone or iPad. It is target at two kinds of users, which are dementia patients and caregivers. It is able to update patient's situations to his/her caregivers. The application is also capable of reminding patients with daily issues based on the daily issue settings, such as take tablets, drink water, do exercise, etc. The anticipated application also provides the patients with GPS based map and emergency button, which means they will not get lost when they are out and will always be able to inform their caregivers at the first time when there is something wrong. Although the application could be impeccable as a dementia patients' personal assistant, dementia patients might get confused and even forget to use the application or lost their devices. Which means, the application should also be capable of updating daily issue progress with caregivers and alarm them once there are anomalies. The software might also enable patients and caregivers to chat with each other. The anticipated user interface is multiple pages for different functions and user can switch them via a menu bar at the bottom.

As previous sections mentioned, the final product almost fits all the anticipated features. The only incomplete functions are automatically alarming and online chatting. Except these two functions, all the major functions are realised. Also, there are several additional features apart from the original design, such as memo system and address book in map system. These additional features made the application become more powerful. Overall, we can consider the completion of this project as a success.

The second strength of this project is the efficiency of the product. As already mentioned, there are several design considered to improve the efficiency, such as local grammar checking before checking with server. Also the database were carefully designed, the data duplication has been avoided. Most SQL queries involved multiple tables were queried by *Table Join* function, which made queries become more efficient.

Thirdly, the user experience had been always considered during the entire implementation. The auto-login as well as different meaningful ringtone were intended to deliver a better user experience.

Also, another strength of this application is the robust and reliable performance. For patient users, the user can still use daily system, memo system, location system as usual, even if there is no Internet access. On the other hand, the database will not be damaged by updating failures, since there are PHP pages as handler between the application and remote database.

Lastly, the feedbacks from supervisors and myself user experience were seriously considered in the entire implementation phase, which improved the acceptance of the application. Based on discussion with my supervisors and background researches, the purpose of the project was extremely clear. The application aims to fill the market gaps in dementia aids, especially in the next decades. Because current dementia patients may not used to use a smart phone, but the young generation who are addicted to the mobile phones will eventually grow old [10].
The features of the application were also based on researches and discussions with supervisors. Overall, the software can be classified as a success, in terms of meeting project requirements.

6.3. Weaknesses of the Solution Produced

As mentioned in previous section, there are two features missing from the specification, which are automatically alarming as well as online chatting. The auto-alarming feature needs a special algorithm to determine what situation should be regarded as an anomaly. This could be done in future implementations. The online chatting function is not done yet, but actually the cancellation message function in alarm system is already be able to deliver instant message now. If this function is valuable to this software, I will add it in the release of next version.

Secondly, the user interface for dementia patients may not powerful enough yet. Although I tried my best on the interface design, it still needs extra improvements, since the old people may suffer from reading and interacting with general designed mobile applications. More

researches on user modelling and interface issues in the medical domain is needed. Maybe employe a voice-over to speak items on the screen will deliver a better user experience for dementia patients. Enabling users to adjust text font size may also be helpful. These functions could be considered in further developments.

7. Learning Points

This project has provided many learning points in key areas.

My problem solving skills were tested throughout the implementation of the project. The entire system is a huge challenge for me, since I had never implemented such scale systems before. It is also my first time to conduct a project on my own. During the process, I learnt how to break a huge system into several subsystem and difficult problem into several parts. After this implementation, I am not afraid of big project any more, since I knew I can divide and conquer it.

As I already mentioned, I am new to Objective-C programming as well as several other skills, I had to spend a large amount of time on learning before coding. In order to complete the project on time, I had to manage my time and workload. The time management and task prioritisation skills were improved through the implementation. These two skills will benefit me a lot with future learning and working.

During the course of the project, my technical skills have also grown because of the different programming languages and technologies that I have used. After coding thousands of lines, I think I am a skilled Objective-C developer now. Also, I now have working knowledge of PHP, MySQL, SQLite, JSON and APNs. PHP and MySQL, although I did have prior knowledge of these technologies, I used them to produce things I had never done before, thus I did expand my knowledge of them, such as using JOIN function in MySQL. SQLite, JSON and APNs were technologies I had never used before, thus learning them alongside carrying the project was not easy. Through learning these, I knew more things about implementing an iOS app, such as reading data from remote database, storing data locally, creating a number

of credentials for Apple developments. Now I feel that I can build upon these techniques and learn to produce much more advanced products.

Since I decided to employe a remote backend for this application, I gained a lot of knowledge about how to use a server for hosting purpose. I not only learnt how to use FileZilla to manage files on the server, but also how to use PHPMyAdmin manage the MySQL database. I think these techniques are very important to me, since I am also willing to write web applications in future.

Although I did not deliver a perfect user interface for dementia patient users in this implementation, I still gained a lot of experience in interface design. In order to improve user experience, I did a number of researches on UI design and also drew several different interface sketches during the design stage. I read several theory and concepts about UI design, and I think I have gradually formed my own opinion on it.

I also gained self-confidence and presentation skills through the demonstration and presentations. Public speaking has always been a nightmare to me, since I have suffered from anxiety. However, I was fully immersed in this project and I really proud of the final product. These enable me to confidently introduce my design as well as the final outcome. I realised that I could deliver a good presentation if I am very familiar with the content.

Through the project I have gained both skills and knowledge, both technical and non-technical, due to this I have grown and developed my skill set. I believe this experience will put me in good stead for future employment.

8. Professional Issues

There are the British Computer Society (BCS) standards for its members to follow. During the entire implementation of this project, two documents are strictly conformed. One is the BCS Code of Conduct [11]. Another one is the BCS Code of Good Practice [12].

Actually, the Code of Good Practice has been already retired by the BCS since 2011 [13]. However, I still conformed to the standards since it is a project requirement. In regards to the Code of Practice, it states that member should, 'strive to understand the organisation's business and search for changes that will bring tangible benefits'. The entire project has been detailed discussed with my supervisor, and we found out it can fill a market gap of dementia patient personal aids.

The document also states that members should 'check that the code is in accordance with the design specification and resolve any differences'. This principle I had been always followed during the implementation. In order to refactor and further implement, I compared the implementation with specification everyday after coding.

Another statement says, when programming, members should 'make themselves aware of the limitations of the platform (operating and hardware) and avoid programming techniques that will make inefficient use of the platform. I had always be aware that the implementation will only work with iOS devices, which are iPhone and iPad. *Dementia Care* does not support Apple Watch yet, but it might be added in next release. I think iOS the amount of potential user that provided by iOS platform is large enough. Maybe in the future, I will consider about the reimplementation in the platform of Android.

There are a lot more standards of Code of Good Practice, I believe that I have adhered where possible.

Moving on to the BCS Code of Conduct, I had carefully read and followed its requirements in public interest, professional competence and integrity, duty to relevant authority and duty to the profession. For example, it stated for professional competence and integrity that a

member should, ‘only undertake to do work or provide a service that is within their professional competence’. In order to realise the project, I had spent a huge amount of time on learning before implementing. I believe at this stage that it was well with my competence to learn the required technologies and implement the solution. Thus, I was undertaking work which was within my professional competence. Also, under the section of professional integrity the BCS states that,’You shall NOT claim any level of competence that you do not posses’. I checked all the documentations, I did not made any competence claims in this project.

The Code of Conduct also states that members should avoid conflict with relevant authority. I do not believe any negative altercations took place during any period of the project. There also is a standard that members should ‘respect and value alternative viewpoints and, seek, accept and offer honest criticisms of work’. During the entire implementation, any constructive criticism that I received from feedback was taken on board.

After carefully consulting all of the standards stated in the BCS Code of Conduct and Code of Good Practice documents, this project as well as the way it was conducted satisfy all the requirements.

9. Bibliography

1. K. Langa, DA Levine: The diagnosis and management of mild cognitive impairment: a clinical review. The Journal of the American Medical Association, 312(23): 51-61, 2014.
2. J. Wang: Relationship between ageing and aged diseases. Chinese Journal of Basic Medicine in Traditional Chinese Medicine: 12-15, 2000.
3. C. Cunningham: Understanding challenging behavior in patients with dementia. Nursing standard, 20(47): 42-5, 2006.
4. Q Care Ltd. Top 5 Apps For People Living With Dementia. http://www.qcare.co.uk/top-5-apps-for-people-living-with-dementia.html. Accessed May 7, 2016.
5. Apple App Store. Dementia. https://itunes.apple.com/gb/. Accessed May 8, 2016.
6. Alzheimer's Association. Alzheimer's Caregiving - There's An App For That. http://www.alzheimersblog.org/2013/08/19/alzheimers-caregiving-theres-app-that/. Accessed May 8, 2016 .
7. iOS Developer Library. Apple Push Notification Service. https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html. Accessed May 8, 2016.
8. B. Berger: Mobile Navigation. Mobile Growth, 2016.
9. T. Wilson: Evaluation strategies for library/information systems. Evaluating Community Programs and Initiatives, 38(3): 23-27, 2010.
10. P. Kendall. Youngsters addicted to mobile phones. http://www.dailymail.co.uk/news/article-45763/Youngsters-addicted-mobile-phones.html. Accessed May 9, 2016.
11. The British Computer Society. Code of Conduct for BCS Members. http://www.bcs.org/upload/pdf/conduct.pdf, June 2015. Accessed May 9, 2016.
12. The British Computer Society. Code of Good Practice. http://www.bcs.org/upload/pdf/cop.pdf, September 2014. Accessed May 9, 2016.
13. The British Computer Society. BCS Trustee Board agrees revised Code of Conduct. http://www.bcs.org/content/conWebDoc/39989, May 2011, Accessed May 9, 2016.

10. Appendices

10.1. Code Listing

Since there are too many code contained in this project. I will simply submit all the source codes in a zipped archive together with this report. It will contains:

- All the source code in Objective-C and PHP.
- All the certifications for Apple iOS development.
- All the icons and ringtones used in this application.

10.2. Unpacking and Installing

Included with this report is a zip archive called ‘Yuyang_FYP_DementiaCare’. This contains all the files mentioned in the above section. The remote database and PHP server are ready for use. To install the application in your own device, just connect your iPone/iPad to the computer, and open the ‘DementiaCare’ folder, which contained in the zip archive. Open the Xcode project file ‘DementiaCare.xcodeproj’ to test and use the application.

10.3. User Guide

1. Launch

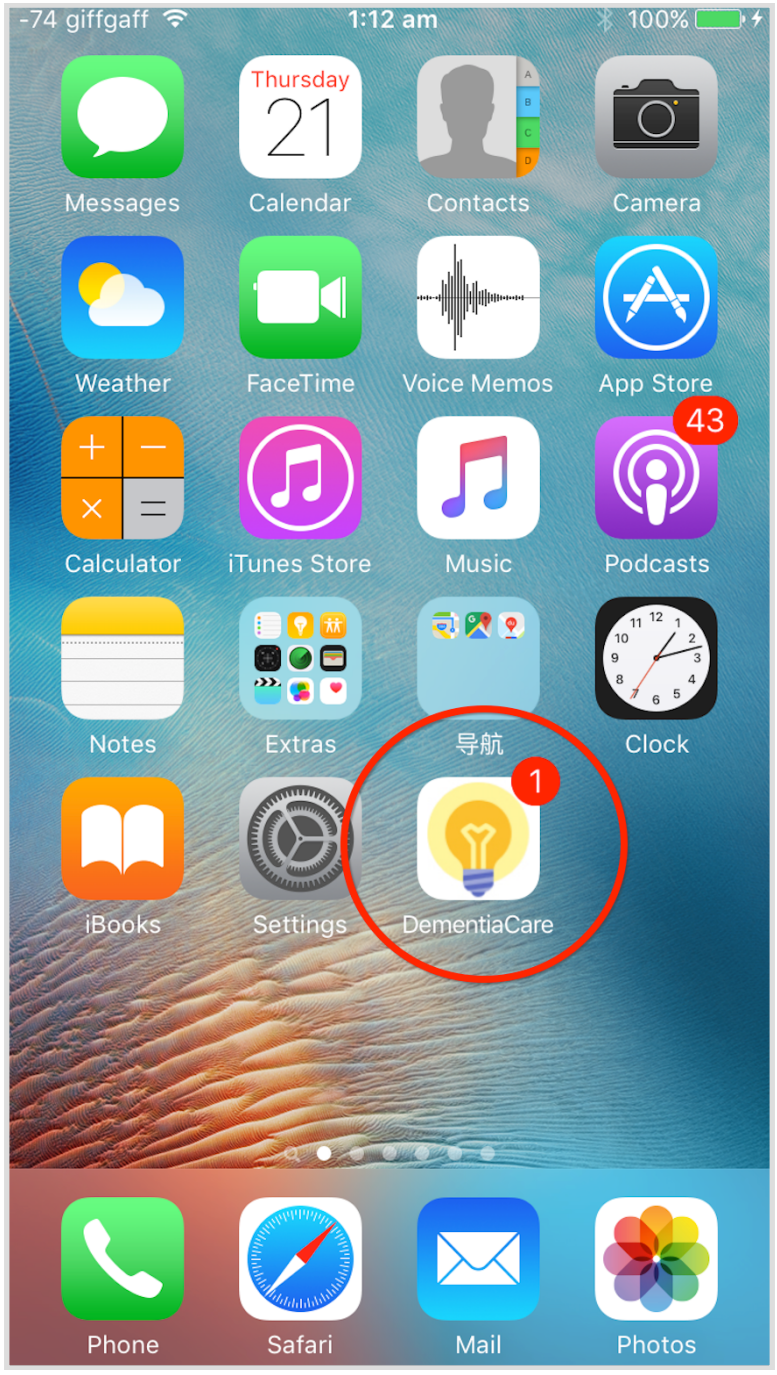

Figure 10-1. Application icon

The application icon as I cycled. You can launch this application by clicking it.

2. Sign in & Sign up & reset password

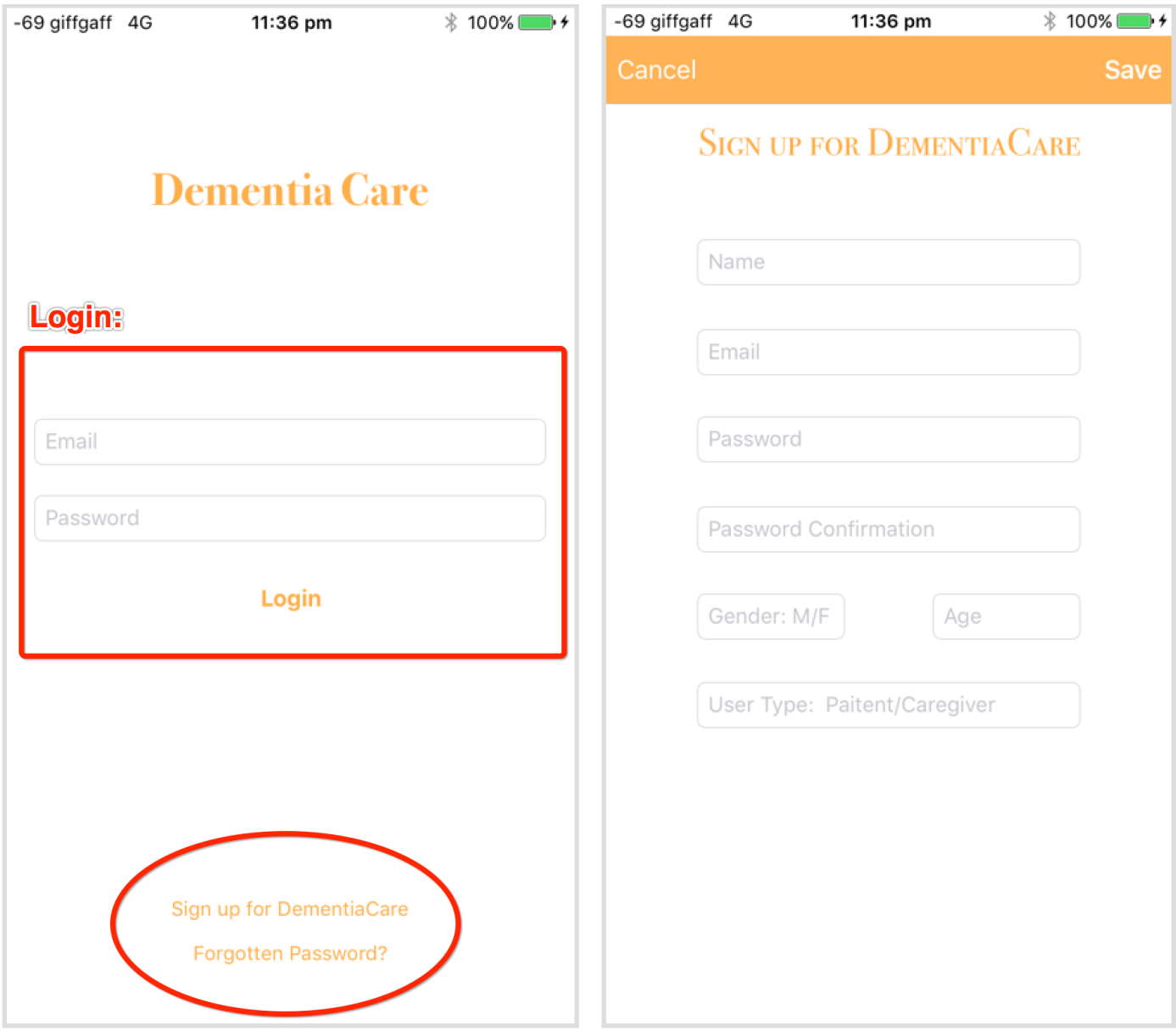

Figure 10-2. Sign in & sign up

Once the you launched the application, you will see the left interface. You can login by entering your login credential in the two cycled textfields. If you are new to *Dementia Care*, you can click 'sign up for DementiaCare' in the cycled area to enter the right sign up view. In the sign up view, you can enter your information according to the instructions to complete your registration.

Once you successful signed up, you will be navigated to the left login page again. Enter your login credential and login now.

3. Use the application (For patient user, caregivers can go to Section 15.)

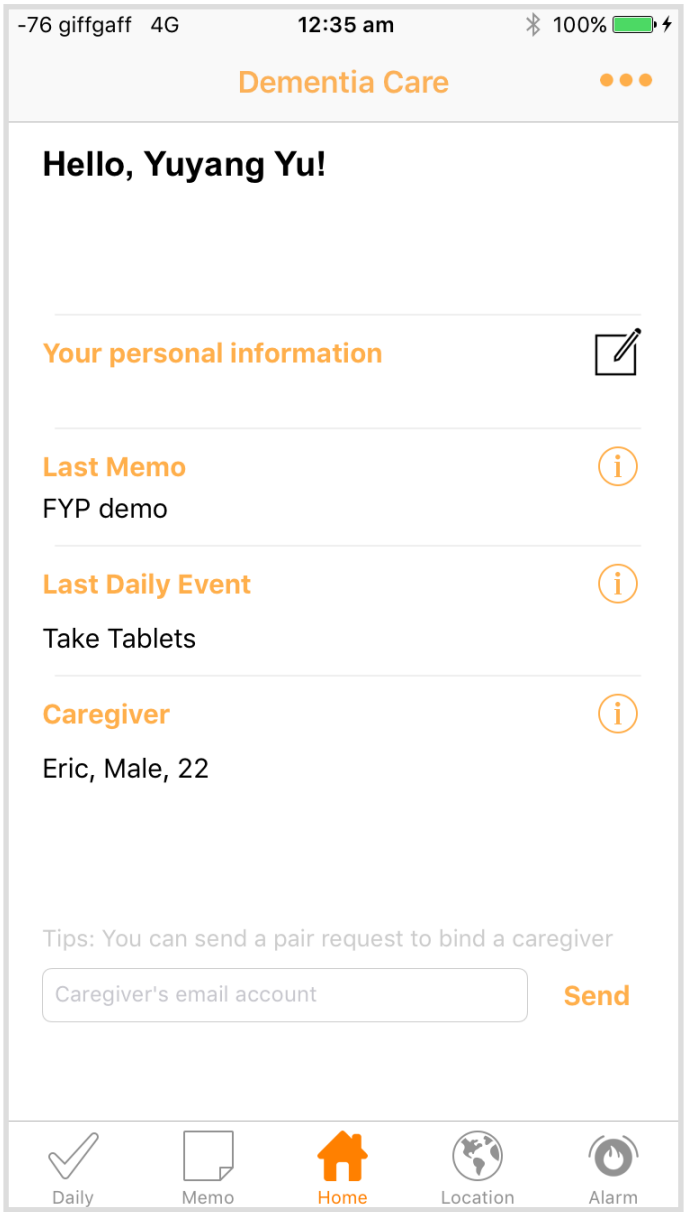

Figure 10-3. Patient home page

If you are a patient user, you will be take to this homepage after logged in. You can use the bottom tab bar navigation to different subsystem of this application. The homepage displays your basic information. You can also send caregiver pair request in this view by entering your caregiver's email account in the bottom textfield.

You can enter your caregivers and requests management view by clicking the detail button in the Caregiver section.

4. How to manage your caregivers

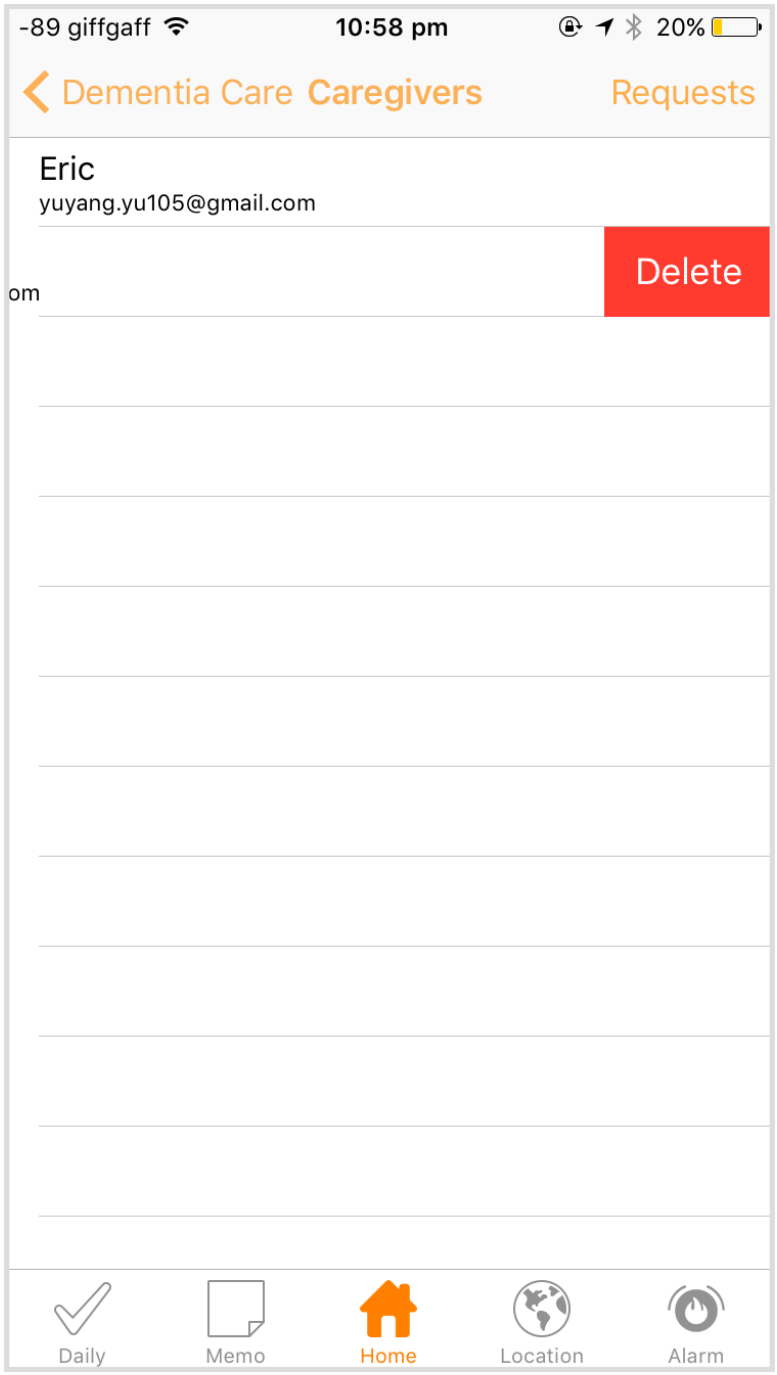

Figure 10-4. Caregiver management

As mentioned above, if you click the caregiver detail button, you will be navigated to this view. There is a table shows all your linked caregivers. If you want to unbind with some caregiver, you can left swipe this caregiver, and click the 'Delete' button as above picture showed.

5. How to accept/refuse link requests

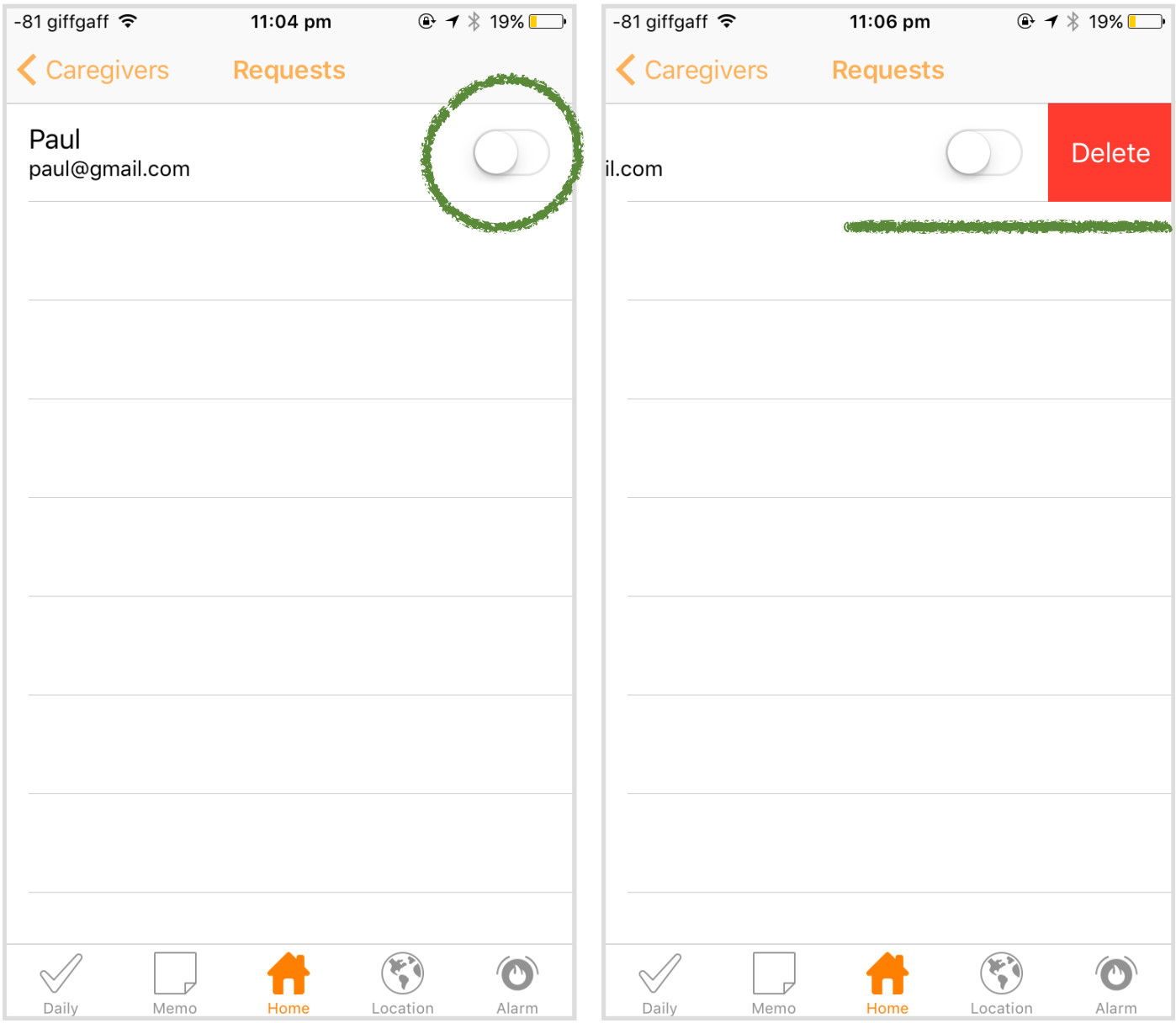

Figure 10-5. Request management

You can also handle linking requests by clicking 'Requests' on the top right corner in previous Caregiver view.

In this view, if you want to accept a link request, you can right swipe the switcher cycled in the left screen shot.

If you want to refuse a request, just left swipe this request, and then click delete. This can be done as underlined in the right picture.

6. Check daily routine events

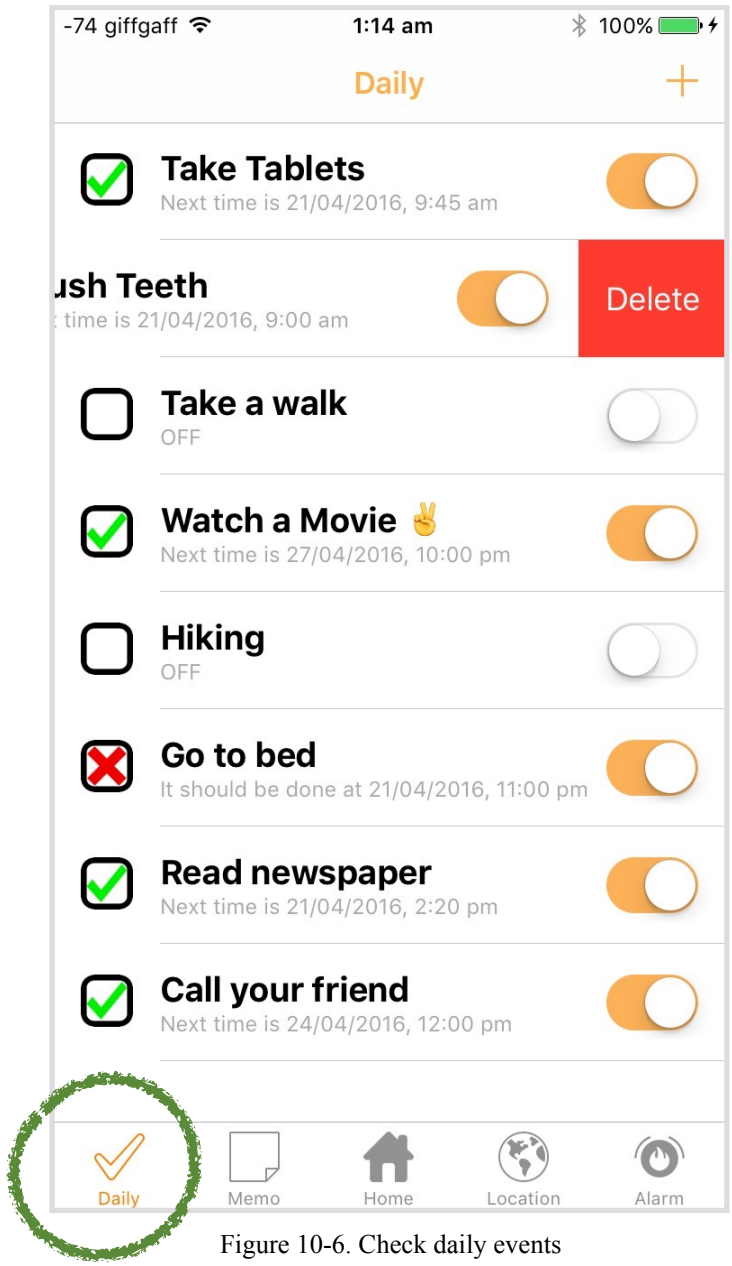

Figure 10-6. Check daily events

To go to the daily routine view, just tap the *Daily* button as cycled. As shown, there are a number of daily routine events. The left side checkboxes has three status: check mark indicates completed, cross mark indicates waiting for completion and blank checkbox indicates the reminder has been turned off. The right side switcher enables user to turn ON/ OFF each events. Moreover, user can delete an event by swiping left and then clicking *Delete*. User can also check one event detail and then update completion by clicking that event cell. The upper-right *add* button enables user to set more event reminders.

7. Set Daily routine reminder

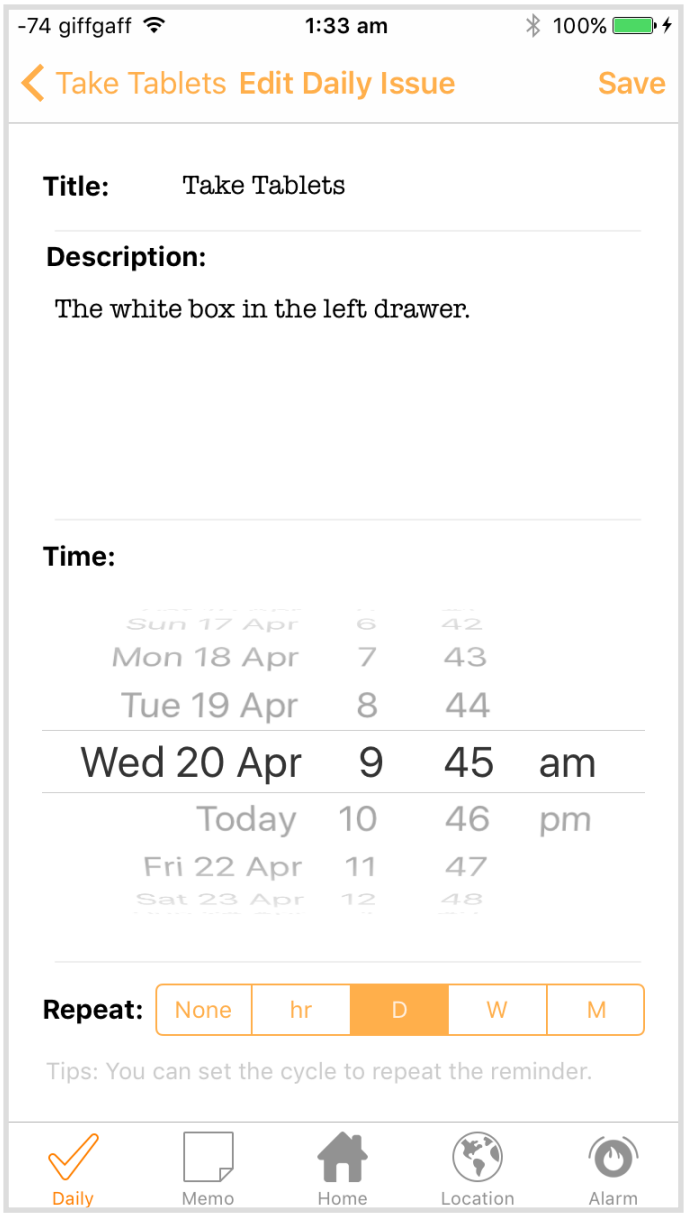

Figure 10-7. Request management

As above mentioned, you can enter the daily routine creation view by tapping the *Add* button. There are title textfield, description textview, time picker as well as a repeat segment scheduler. These kits work together to enable user set up a daily routine reminder as they wanted.

The repeat segment enables you to repeat the event every hour, every day, every week, every month or even never repeat.

8. Time to do something

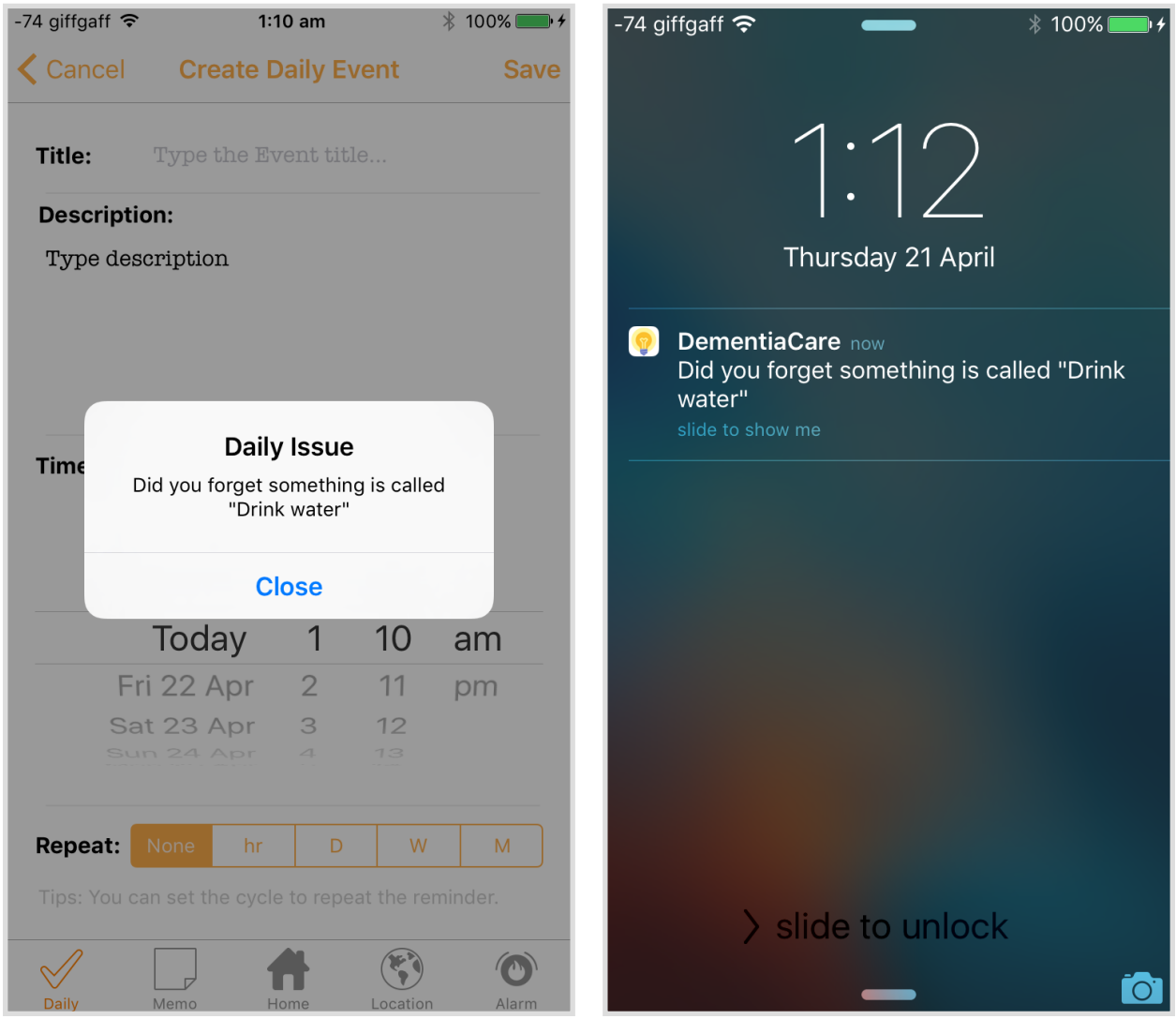

Figure 10-8. Time to do something

When it is time to do something, there will be a reminding notification. You can then go to the daily event list to select the event and then report completion.

9. Report Completion

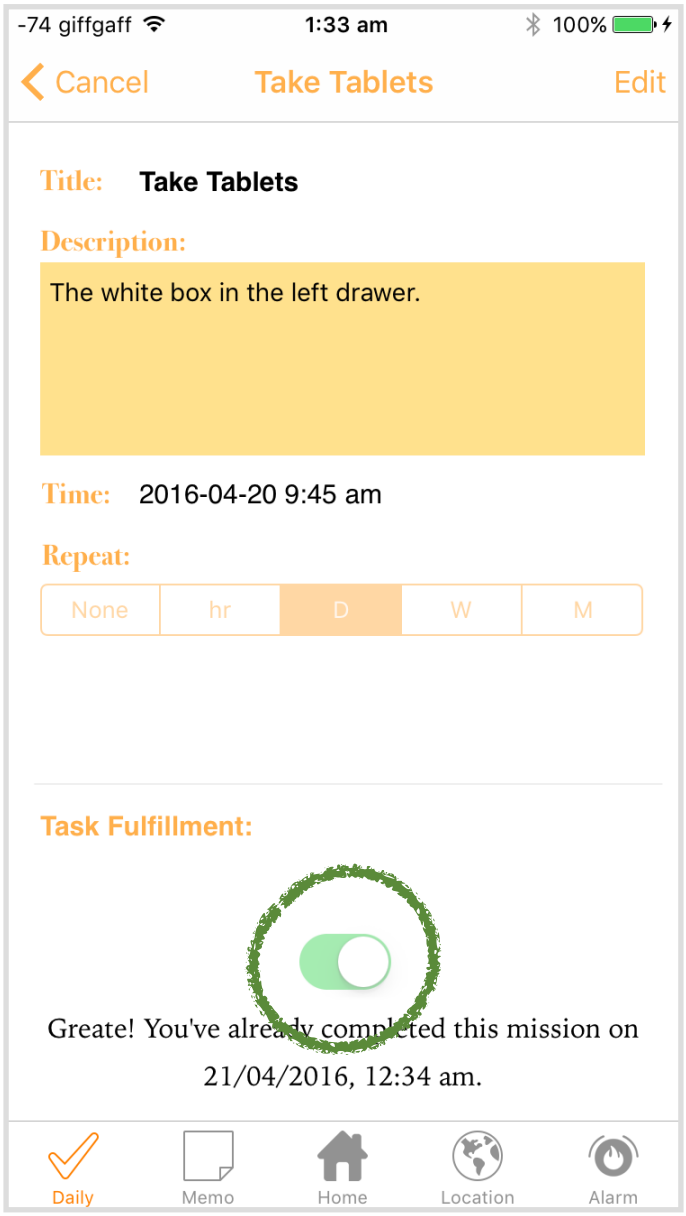

Figure 10-9. Report completion

When you tapped a specified daily routine, you will see its details as above picture showed. You can report your completion in the cycled area. This report switcher will be disabled if the task is already completed or not ready for completion.

10. View Memos

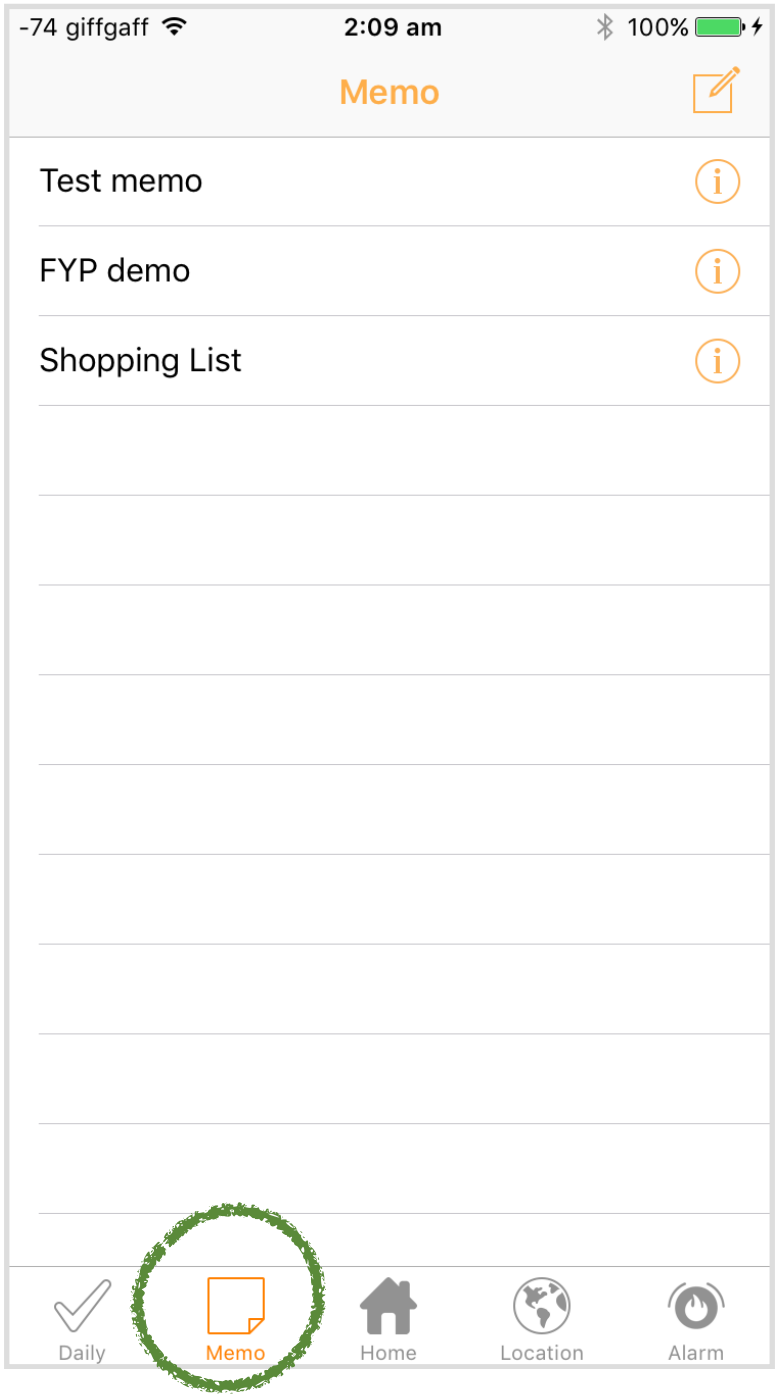

Figure 10-9. Report completion

You can see your memos by clicking the *Memo* button as highlighted in the navigation bar. By clicking each detail button, you can check details of each memo.

You can also create a new memo by click the top right corner *Compose* button.

11. Create Memo

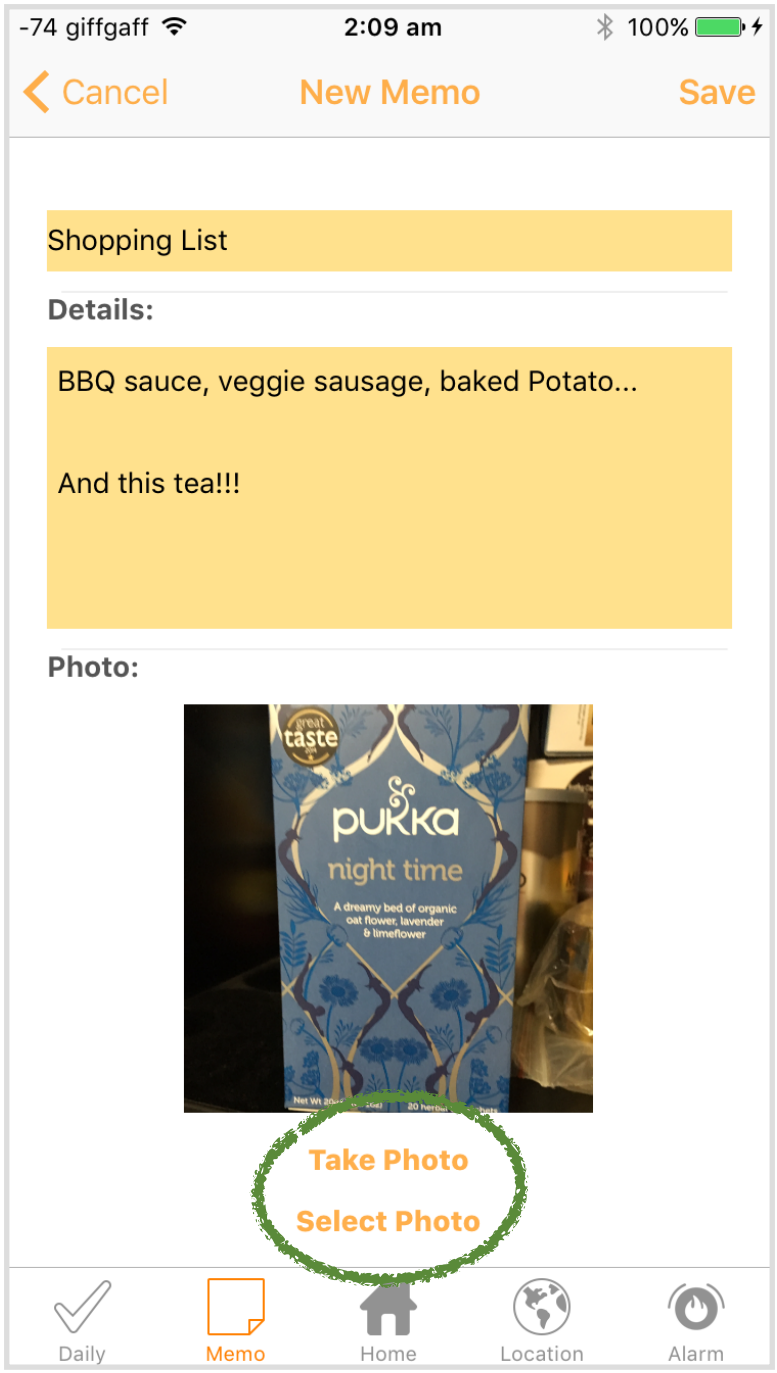

Figure 10-10. Create Memo

As shown, you can create a memo by entering the title and details as instructions suggested. You can also insert a photo into the memo by taking one or selecting one from the system.

12. Enter the map

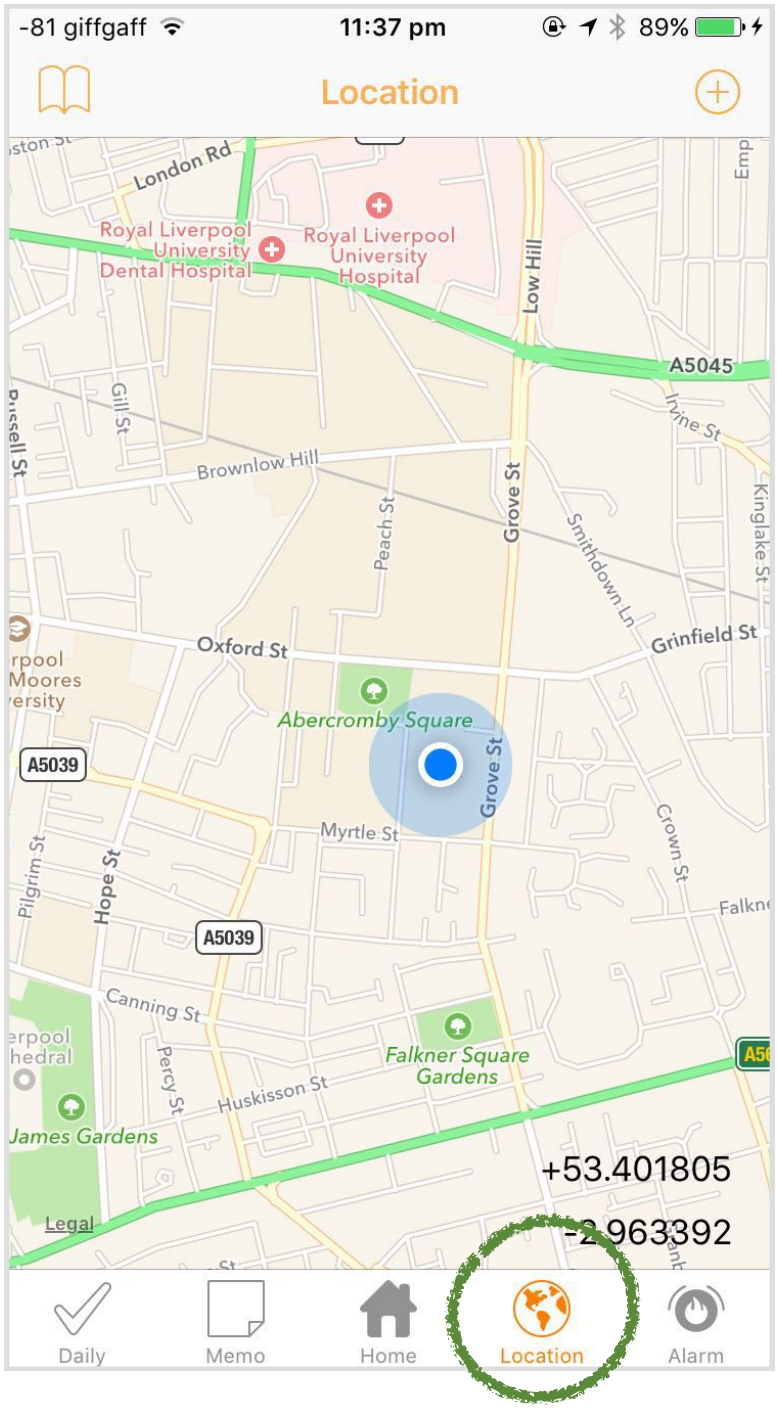

Figure 10-11. Enter location system

You can enter the map view by clicking the *Location* button as highlighted in the navigation bar. You can see your current location here.

You can also add the location into your address book, by clicking the top right *Add* button.

13. Add current location

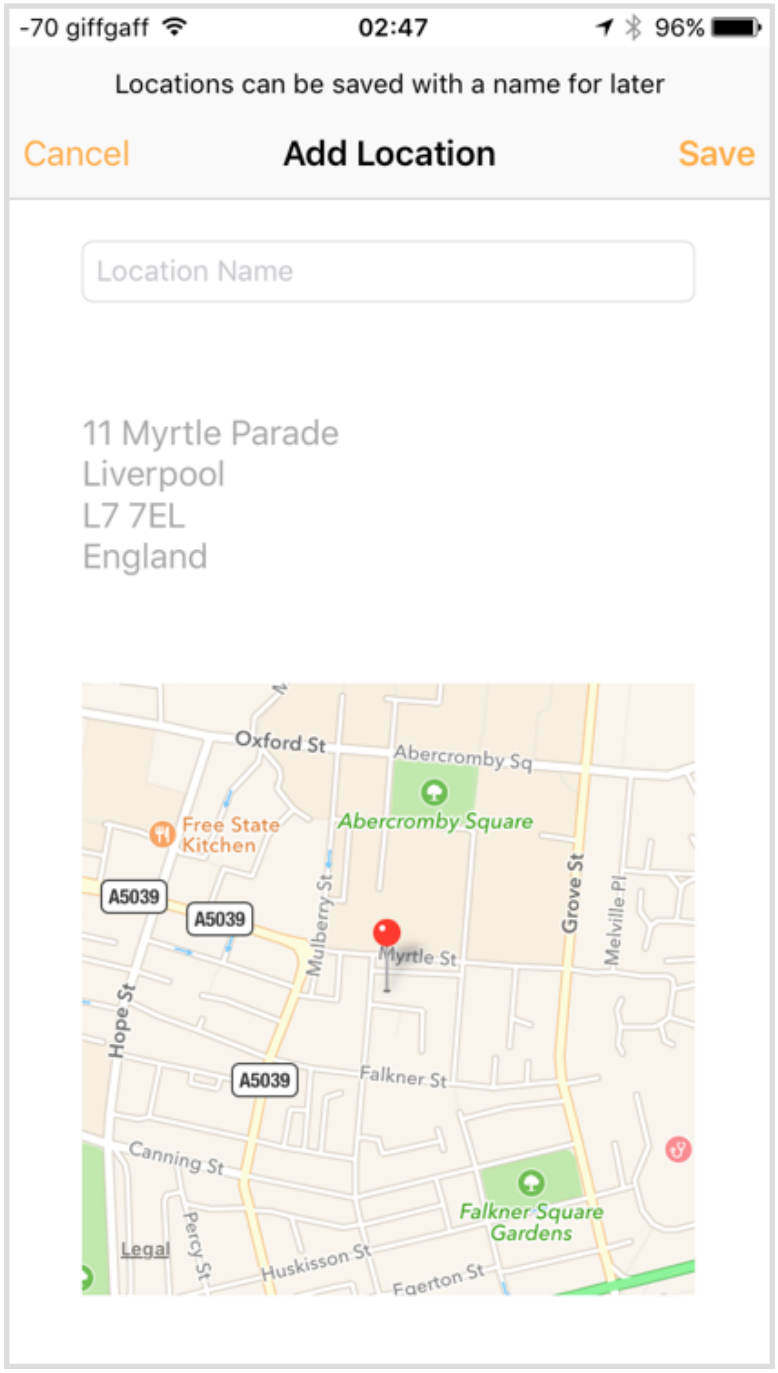

Figure 10-12. Add current location

When you entered the adding location view, you can name your current location and then add it into your address book. Once you named it, you can click *Save* to save it. You can also go back to previous view by clicking *Cancel*.

14. Use emergency alarm

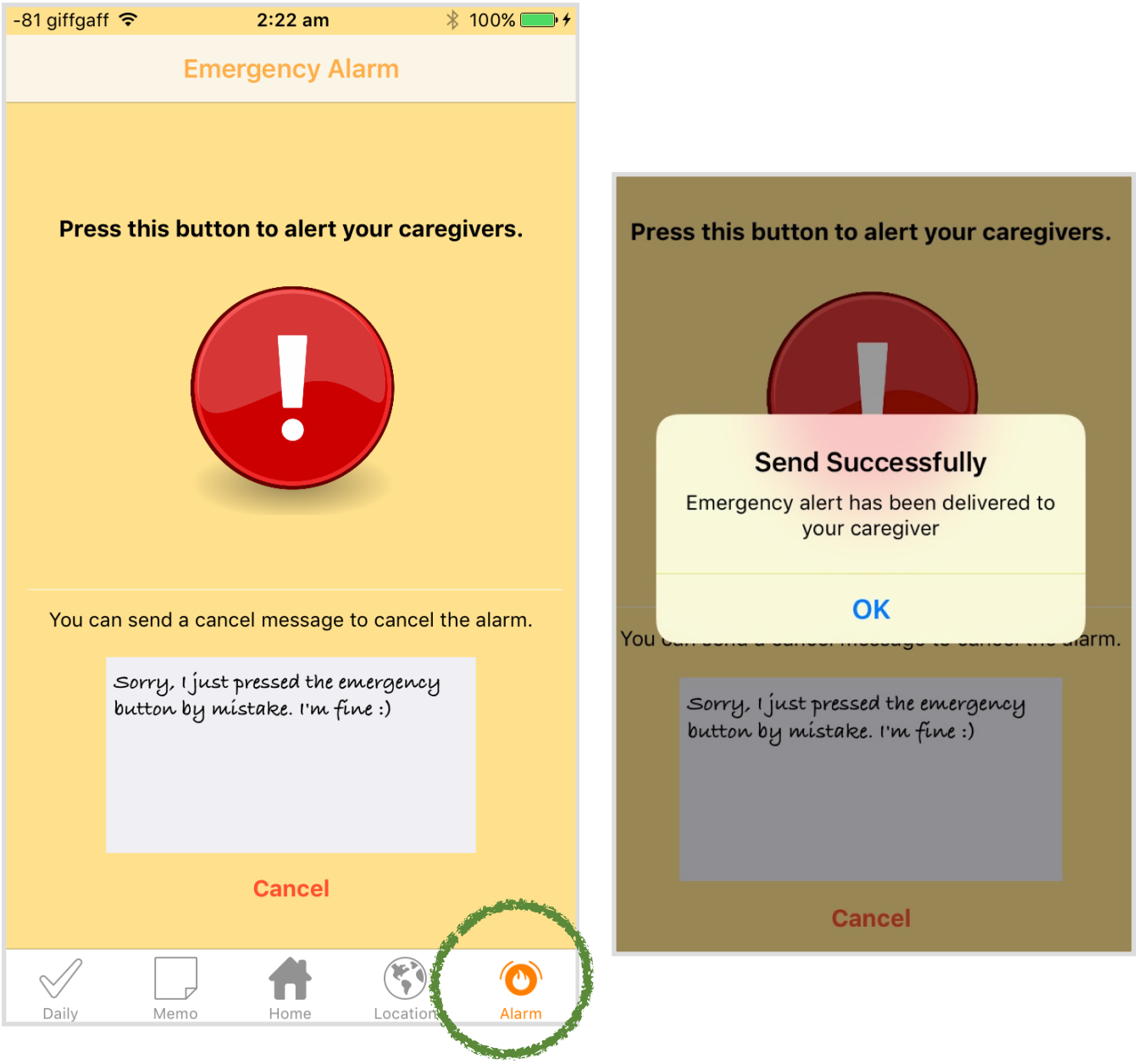

Figure 10-12. Use alarming system

If there is an emergency, you just tap the *Alarm* button as highlighted. Then you can tap the red emergency button to alarm your caregivers immediately. Once the alert successfully sent, there will be a popup window indicates the alarm has been delivered as shown in the right picture.

You can also edit and send a cancellation message to your caregivers, if you are fine.

15. Use the application (For caregiver user, all the patient's usage are introduced above)

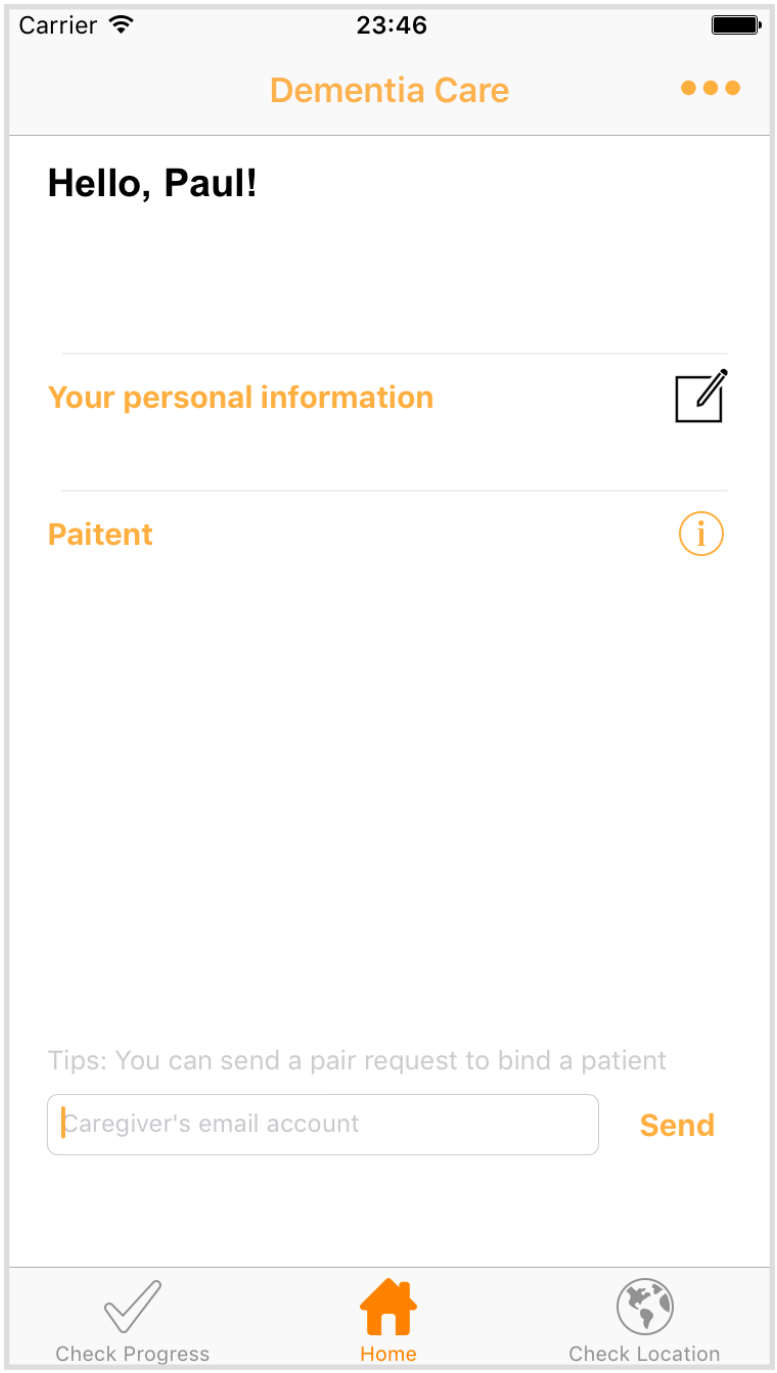

Figure 10-13. Caregiver home page

If you are a caregiver user, you will be take to this homepage after logged in. You can use the bottom tab bar navigation to different subsystem of this application. The homepage displays your basic information.You can also manage your linked patients and linking requests as described in Section 4 and 5.

16. Check Progress

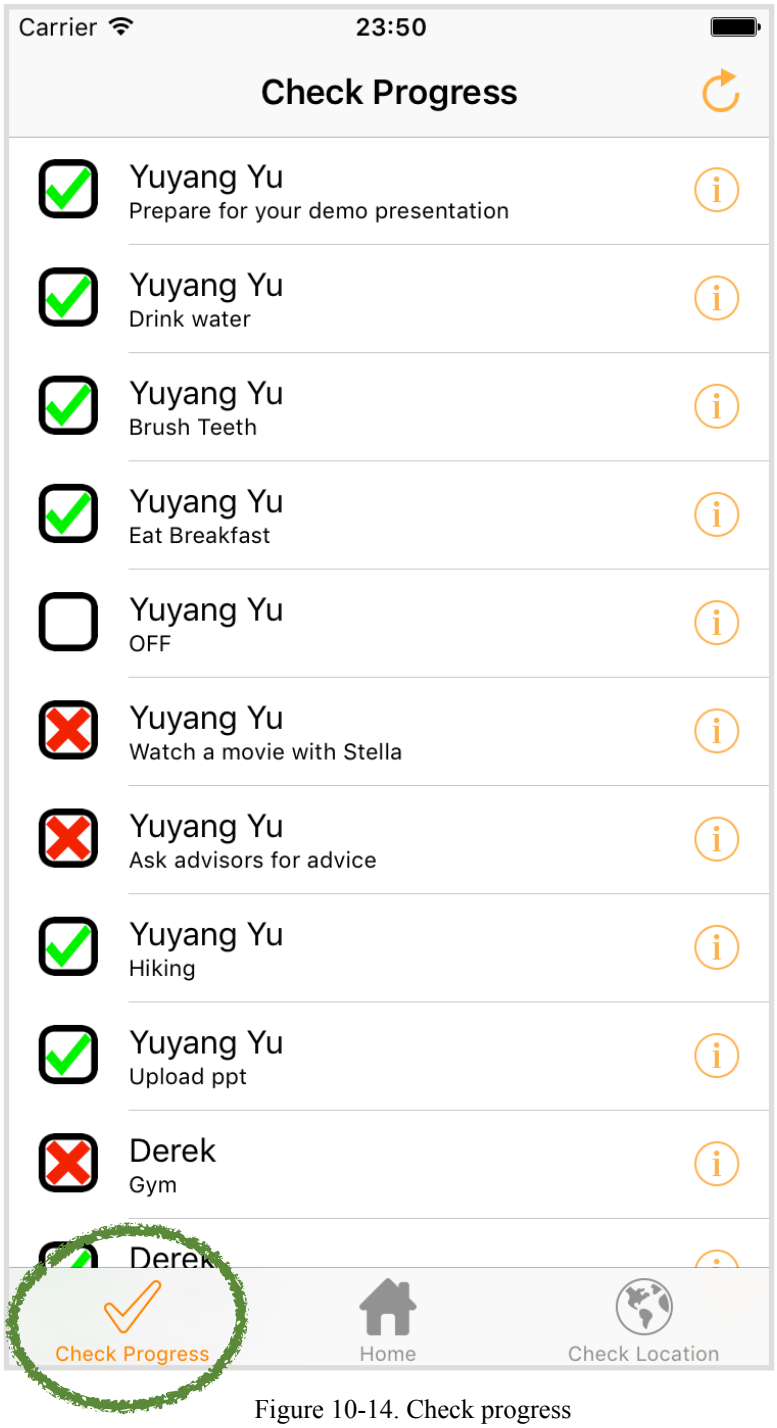

Figure 10-14. Check progress

You can check your patients' daily routine progress by tapping the '*Check Progress'* button. You can see all your patients' daily issues and their completion status. You can see details of each event by clicking the detail button on right side.

The top right corner *Refresh* button, yes, enables you to refresh.

17. Check location

Figure 10-14. Check progress

You can check your patients' real time locations by tapping the '*Check Location'* button. There are multiple map annotations. By clicking one, as above picture shown, you can see who is here and where is here.

Also, there is a *Refresh* button on the top right corner *Refresh* button. Also, it enables you to refresh the map and your patients locations.

18. Log out the system

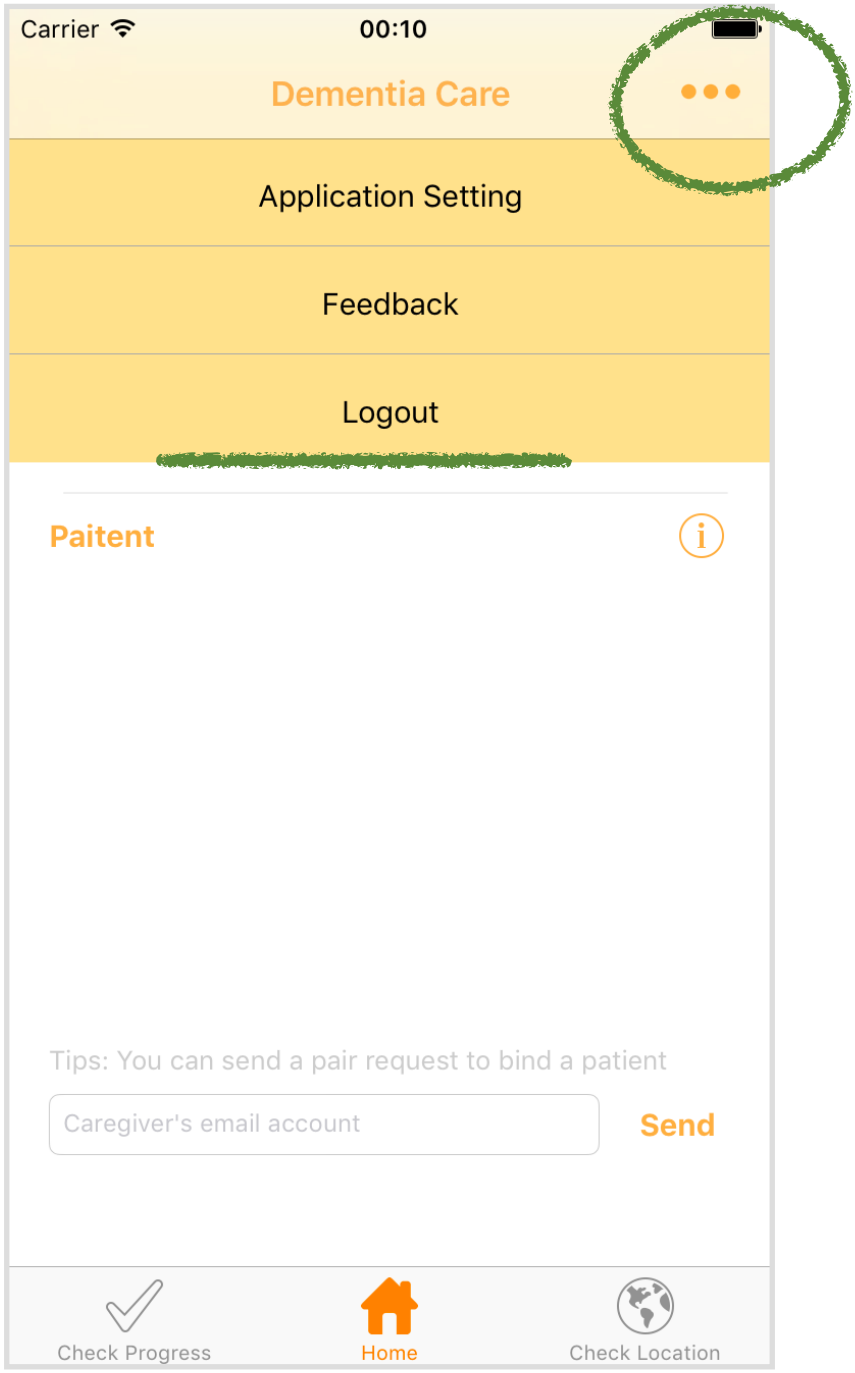

Figure 10-14. Log out

Lastly, you can log out the system by tapping the top right *Menu* button in Homepage. There will be a drop down menu, you can then click *Logout* to logout.