Fintan Guckian
11630129

ITC203

Assignment 1: Architecture – UI – Database Design

NFR	Trigger question	Impact/cost	Benefit	Answer	Strategy	Priority
Secur ity	How secure must the systems data be from outside access?	The system will need security mechanisms .	Comply with strict regulation s.	The system must not allow unauthorised access to medical data at any cost. Breech of data security would be considered a catastrophic failure.	Security certificates will provide secure connections between http servers and user devices	Very important
	Does data travelling between the RTGMD and smartphone have to be secure?			This aspect of the system security will be handled by Sandia.		N/A
	Who can access patient data?	The more people with access the less private the data.	The system needs professio nals involvem ent to function	Patients, Pacific Health Systems doctor/ front line medical staff	Access log table in database keeps track of people viewing patient data.	Important
	How will front line staff access patient data?	Hiring staff to monitor patients 24hours is expensive	This will satisfy a functional requireme nt that patient alert readings be monitore d whenever they occur. It is also more secure as only people sitting at	Pacific health Service will set up a 'control centre' from which a nurse/doctors assistants will receive alerts from patients. The 'control centre' will be monitored 24hrs.	Terminal in an office of Pacific Health Services. Application built and installed only on this terminal. Front line staff can log on and log off at the terminal. Access log tracks data accesses.	Quite Important

			the terminal will be able to access the data.			
	How will doctors access patient data?	We will have to work with existing website	Doctors are already familiar with the Pacific Health Service site.	Doctors will have access to patient data though their Pacific Health Service account which they access via a website.	Build upon existing website.	Quite Important
	Does every doctor have access to the data of every patient in the system or will each patient be assigned a doctor?	In the event that one doctors account is compromised all patient data can be accessed. This is a security concern.	If an emergency is to be handled in the best possible manner the treating doctor will need access to the patients medical history.	Patients preferred doctor may not always be available to handle their emergency. All doctors in the Pacific Health Service will have full access to every patients data.	A doctors account will have access to entire patient database. Accesses can be logged to trace any suspicious account activity. Changes will also be logged for easy reversal of unauthorised actions.	Quite Important
	Who can access patient personal voice recordings and written notes?	Extra feature to toggle privacy	Allows patients more privacy	Patient can opt in/out of allowing access of personal notes for medical staff	Field in the relational database indicating weather personal recording can be viewed by doctors	Important
	Who can set the patient glucose alert level?	Access to this feature must be secure.	The system becomes more flexible as it can adapt to	Only the patients doctor can set and adjust the patients glucose alert level.	An attribute on the patient table of the database schemas that can be altered by the doctor	Important

			the changing health attributes of the patient.		only.	
Perfor manc e	How often are glucose samples taken and sent?	More data to store	More frequent samples means abnormal readings can be discovere d faster	Up to 300 times a day	Discuss server and database options with managed services company.	Quite important
	What language is the Pacific Health Service website written in?	We must conform to the language of the existing website	Easy access for doctors.	JavaScript.	Write website in JavaScript	Quite Important
	Which browsers does the existing website support?	More work for the frond end developers	Better availabilit y of the system	Chrome, Firefox Safari and Internet Explorer	Web page implemented in JavaScript with adjustments for each browser. Web server must query browser	Less important
	What screen sizes does the existing website support?	Our additions must be viewable in these screen sizes	Consisten t website	The existing website is compatible with standard PC screens, tablets and smartphones	Web server to query browser to find out what kind of device is being used. Code to be adjusted for each screen size.	
	What OS should the front line terminal use?	Will have to purchase Windows OS	Staff will not need training to navigate around the OS	Windows would be best as it is the most widely used and our staff likely already be	Use Windows	Less Important

				able to navigate around it.		
	How long is acceptable downtime?	Backup devices will cost extra	System will be more reliable	While we recommend patients keep a backup glucose testing device(such as a finger stick) we know some people will not. More than two hours is unacceptable.	Could consider a backup server in case the primary one fails. Backup database may also be necessary. Will discuss with managed services company.	Important
	What will happen if a persons phone goes dead?			A patient is responsible for keeping their phone on and charged.	Suggest patient bring portable charger with them.	N/A
	What happens if the phone is in a special condition such as flight mode?			The patient should switch to their secondary device in cases like this.		N/A
	What is the distance and transfer of Bluetooth required?	Wearer must keep phone within 100 meters for the system to function.		Sandia will implement using Bluetooth 3 technology enabling data transfers between two devices up to 100 meters apart.		N/A
Response time	How quickly must the system respond to an abnormal glucose reading?	Server must be constantly active without downtime	Quick reaction better medical outcome	Within seconds	Discuss server options with managed services company. Backup server may be required.	Quite Important

Capacity	How much data will the server be receiving simultaneously from patients?	More storage capacity and more powerful servers = more money	Keep system working reliably	To begin with our system should be able to handle up to 1000 patients and be readily expandable. 1000 patients can each make up to 300 data entries daily.	Large database needed. Will be expensive to store and manage. Powerful server also needed. Both should be scaleable. Discuss options with managed services company.	Quite Important
	What is the predicted increase in users expected be per year?	Hardware must be scalable	System not constrained by initial implementation.	We hope to grow to service more than double the number of users in the next five years.	Scalable databases and servers	Less Important
	How long does patient data need to be kept for?	May need to increase capacity as the data bank builds up.	Patients have a better service	Patients should be able to view up to a years worth of statistics.	Scalable databases.	Less Important
Availability	What is the range this system should cover?	System dependant of patients ability to connect to the internet		The system should be equally available all over NSW	Recommend patient acquire 3g or 4g mobile connections.	N/A
	What happens if the user is not connected to the internet?			The user is expected to keep an active internet connection.		N/A
	What is the runtime environment of the web server that hosts the existing	Our web server will have to communicate with the existing server	User unaware that website is sourced form multiple	Apache		Quite Important

	Pacific Health Service website?		web servers			
Reco very	How often should the data be backed up?	Backup will require more storage space.	Can recover data in the event data corruptio n or hardware malfuncti on	Once a day	Backup database.	Important
Relia bility	How important is the saved data	Backup will require more storage space.	Problems with one server or database and the second can take over/ save the data.	Peoples medical records must be kept properly in order to aid medical treatment and for research purposes.	Backup database.	Important
	How will glucose readings be analysed?	Back end must have constant contact with patients application.	Data processin g occurs in one place (back end).	The application developed by Sandia will receive the reading from the RTGMD and relay onto the system. It will be analysed by an application in the back end and if an abnormal reading is found to have occurred the back end will send a warning to the device to trigger an alarm	Application on application server can test readings against alert level set by doctor (which is stored in the database) and trigger a warning.	Important
	What is an	Can be	Reliable	Any more	Managed	Important

	acceptable window of time to get the system fixed if it goes down?	expensive to call someone last minute to fix a problem	system.	than an hour is unacceptable. Patient should be alerted if their data is not being monitored.	services can look after hardware problems. Our company can be called in to fix software issues. .	
	How should a the user be warned if the system is not working properly?		Allows the user to take control of the situation when the device fails	The device will alert the patient that the system is not responding if it does not receive 'data accepted' signals from the web server. The alert will be similar to that of a high/low glucose reading. This is a feature of Sandia's application.		N/A
Compatibility	Will this be compatible with all smartphones?	Users with phones that run other operating systems may be upset about having to buy a new phone to use the RTGMD	Less cost involved in designing software for only on e OS.	At the moment Sandia are developing an application that runs on Google android only		N/A
Maintainability	Will the server be on site or will you rent server space?	Servers not owned by the stakeholders	No need to have tech staff to maintain 'home' servers.	We are looking into managed server options. We would like an outside service to run and manage our servers for	Managed services	Quite important

				us.		
Usability	How often can the user remove the device?			The device will ideally be worn always but can be taken off. If the patient must remove for more than a couple of hours we recommend they use another glucose testing device.		N/A
	Are there special requirements for old people?			UI should cater for all levels of computer literacy with options for making text large. This is part of Sandia's application.		N/A
Cost	Should less expensive hardware be preferred if it meets functional requirements but is less optimal?	Could be problems if the system expands and better performing hardware is needed later on.	Keep within a budget	As long as data is secure and the system functions within the specified requirements, less expensive hardware should be opted for	Use older reliable technology	Important

Question 1: Architecture Notebook

Goals and Philosophy

The most important considerations for this system are to do with security of data as it traverses the system and reliability of the system. Protection of patient data is vital we must stay in line with strict Australian medical data legislation. Breeches of this legislation can lead to heavy fines. Protecting data is also vital to the reputation of Sandia Medical devices. A breech of data security could cause damage to the reputation of the company.

Reliability of the system is vital so that patients wont be put in any medical harm by failure of the systems components. Patients are relying on this system to inform them when their glucose levels are too high or too low so that they can take the appropriate action. Failure to accurately notify a patient of their levels of glucose can have severe medical consequences. The patient will rely on contact from a medical professional in the event of a possible emergency. They will also rely on the system to tell them when the system itself is not functioning. In this case they need to switch to a secondary glucose monitoring device. Responsibility for the correct functioning of the RTGMD and the smartphone application itself is with Sandia.

Cost has been identified as another important constraint. Affordability should be prioritized except over security and reliability. Use of older technologies is acceptable as long as it does not undermine security or reliability.

Ease of implementation is also a factor. Decisions about software and hardware will be influenced by the familiarity of the development team to them. Of course, if an unfamiliar development tool is seen to improve security, reliability or reduce cost it should be favoured.

The stakeholders also wish to have a high level of usability that can cater for all members of the computer literacy spectrum. The app and web page should be simple to navigate.

With regards to performance, the system need not be any faster and should be no slower than a standard application/web page.

Considering capacity, the system should be scaleable to allow for more users and to hold the backlog of existing user data as it grows over time.

The system should be available to anyone in Australia with an internet connection.

In terms of maintainability, the system servers should be maintained off site by an independent organization. The stakeholders do not want the responsibility of maintaining and fixing servers.

Assumptions, Constraints and Dependencies

Assumptions:

- The RTGMD can use Bluetooth to send data
- The RTGMD will take accurate glucose level recordings
- 24hr staff will be maintained to monitor the system and call patients that record dangerous readings
- The patient will always have a secondary glucose testing option in case the system fails for whatever reason.

Constraints:

- The system must work with a smartphone running Android OS
- The system must work with the existing Pacific Health Service website

Dependencies:

- The patient has a smartphone capable of hosting Android OS
- The patient has a smartphone capable of Bluetooth
- The patient has a smartphone capable of audio recording
- The patient has a smartphone capable of emitting an alarm sound
- The patient will take responsibility for keeping their phone charged, active and Bluetooth active.
- The patient will keep their phone on or close to themselves.
- The patient will be available to answer phone
- The patient will have a constant connection to the internet with limited interruption
- The patient must wear the RTGMD constantly
- The doctor can navigate a web page
- A third party will be responsible for server storage and maintenance

Architecturally Significant Requirements

The RTGMD developed by Sandia uses Bluetooth to send and receive data to an application they built for smartphones running Android OS. Data is to be sent continuously from the RTGMD to other parts of the system for permanent storage and analysis. The patient needs to be informed of the nature of abnormal glucose readings as they are identified. The patient must be able to record written and voice messages in the event of an abnormal glucose recording. The patient must be able to view historical data. The patient must be able to update personal recordings. The patient must be able to choose weather their written and voice recordings can be seen by their doctor. Our system must interface with Sandia's application to satisfy these requirements.

The patient and the patients doctor must be able to view historical data in graphical or raw data form. The patients doctor must be able to make changes to the glucose level setting that triggers an alert. These changes must be logged by the system including who made the changes, old alert level and new alert level. The patient must be able to make changes to their historical recordings at any time. These activities necessitate the need for data storage and for an interface between user and data storage.

The doctors will access the patient data via their existing accounts on the Pacific Health Service website. This means we will have to work with existing code. Doctors must be able to access patient data and adjust 'alert' level from the website.

Front line staff need to be informed about high glucose readings as they occur. The system needs to communicate the nature of the reading and the contact details of the patient. This will require an interface between the front line staff and the RTGMD data readings.

The system is initially required to store data of up to 1000 users. The system must also be ready to scale up in the event that the number of users increases beyond the current capacity. The hardware used for data storage must also be managed outside the company. The data stored by the system will need to be accessed remotely by both patients and physicians. These requirements will effect our data storage decision making.

Medical data is subject to strict privacy laws. Patient trust in the system is contingent on their faith that their data will only be accessed by people authorized. Data of this nature can be targeted by hackers for the purposes of blackmail. For these reasons our system will need security mechanisms.

Patients will trust the system to keep them informed about the state of their blood sugar. Patients will also rely on the system to warn them if they are entering a critical condition. Patients will rely on the system to inform them if the system is not working properly so they can switch to their secondary glucose monitoring devices. Our system will require features to satisfy these reliability concerns.

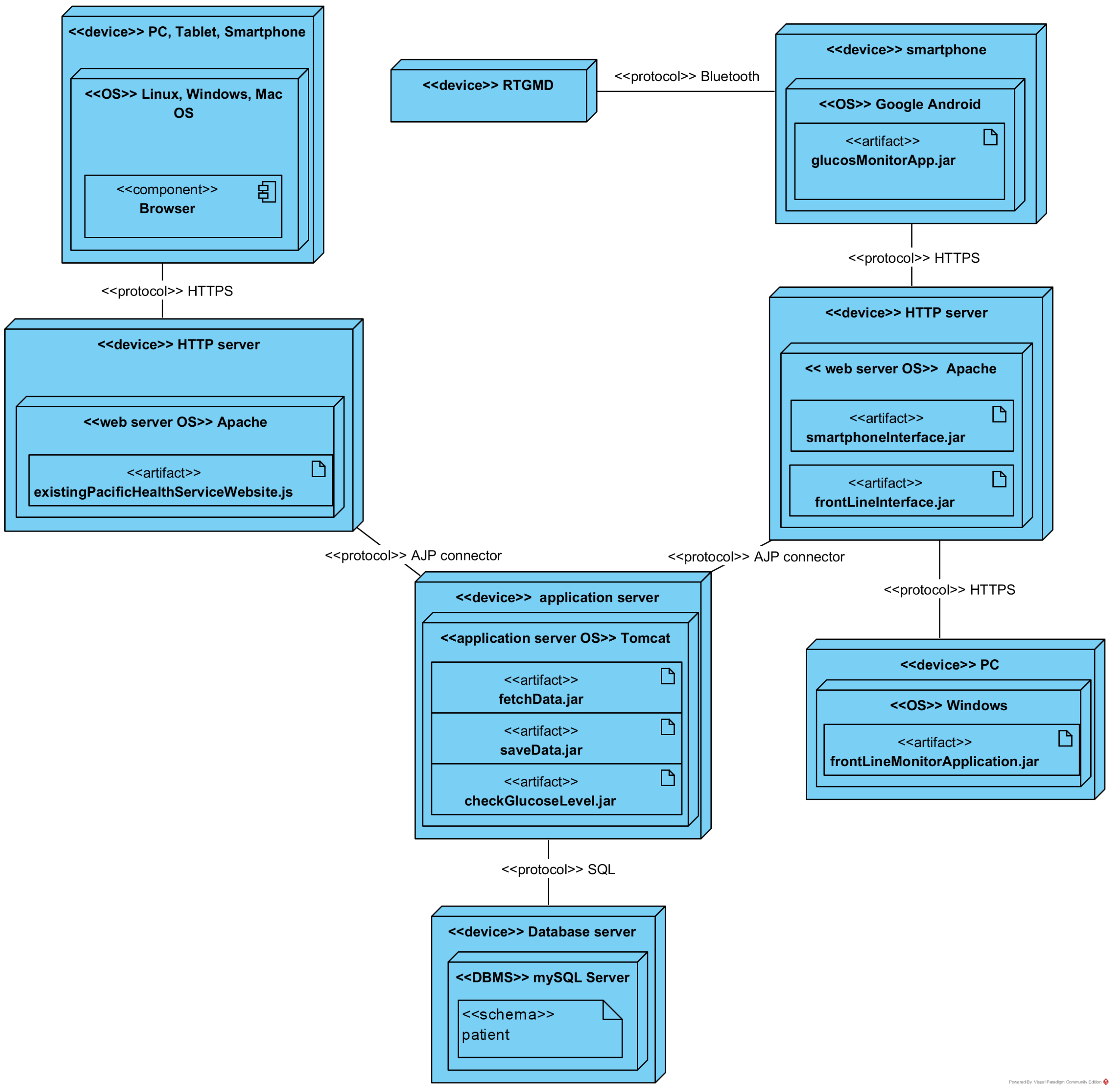
Decisions and Justifications

- All applications will be written in Java as this is the language our team is most familiar with. This will contribute to ease of implementation. Web pages will be coded in JavaScript as this is the language that the existing Pacific Health Service website is written in.
- An android OS application will be written for the patients phone by Sandia. This application will be capable of receiving data from the RTGMD, sending data to a web server and requesting data from a web server.
- Front line staff will be able to monitor the patients using a Java application. The application will exist only on one or two computers in an office of Pacific Health Service. Staff will monitor the patients from here. This set-up limits access to patient data which will help keep data secure. The application server will have a program that watches for abnormal glucose readings. When the Java program finds an abnormal reading it fetches the patients contact details from the database and sends it to the front line staff member manning the desk. The front line staff member can then make the decision weather or not to call the patient with medical instructions. Alert observations automatically store the ID of the staff member that receives the alert. The application will connect to the back end over the internet through a

web server using HTTPS protocol. The internet connection will be secured using security certificates.

- The Pacific Health Service web site will be used by physicians to view patient data. This website is written in JavaScript. We will add a 'Glucose Monitor Data' button to a menu on the home page which will link to a page that allows doctors to view data of all patients on the system (graphs/raw data) as well as the recordings that patients have opted to share. The doctors will be able to adjust 'alert' level glucose through the website. Our web pages must be compatible with Chrome, Firefox, Safari and Internet Explorer as these are the standard web browsers specified by the stakeholders. We will connect our application server to the web server hosting the Pacific Health Service website. We will use AJP connector as the protocol as the web server hosting the Pacific Health Service website uses an Apache runtime environment and our application server uses Tomcat.
- Communication between patient smartphone/front line application and the web server will be implemented using a HTTPS protocol secured with a public key certificate. This will allow patient medical data to make its way around the system securely.
- A web server with an Apache runtime environment will be used to handle requests from the patients smartphones and the front line application. Apache is a free software which will help keep costs down.
- An application server will interface between the database and the various users. An application server can run programs and fetch data requested by the users from the database. This will include data for graphical display, patient contact information (for front line staff). The application server will have a program that tests each glucose reading against the alert level and triggers a warning if the level is too high. The runtime environment for the application server will be Tomcat as it is free.
- A relational database will be used to store the systems data. A relational database is simple to build and maintain and our team has experience with them. This contributes to ease of implementation and maintainability.
- Data will be stored and served using managed services. This takes responsibility for maintaining server equipment away from the stakeholders. This also allows more computing power to be added without stakeholders needing to concern themselves with the implementation. A terabyte capacity will get us started.

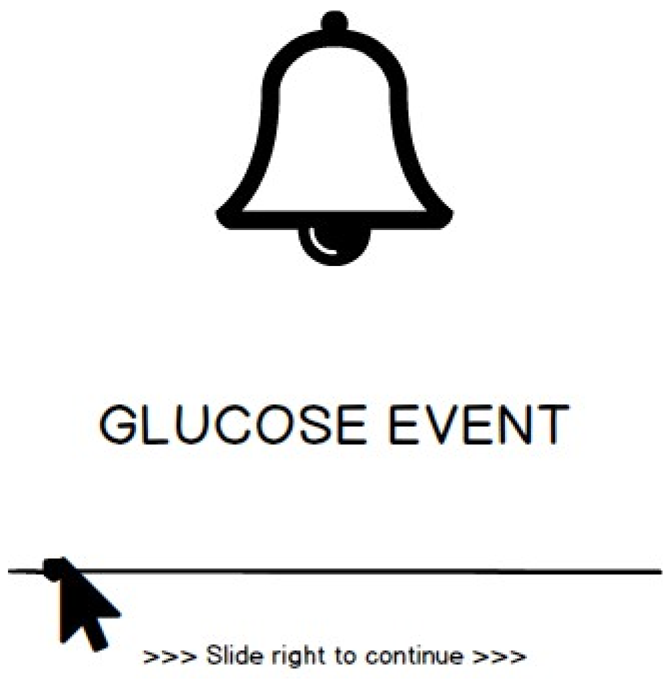
Deployment Diagram

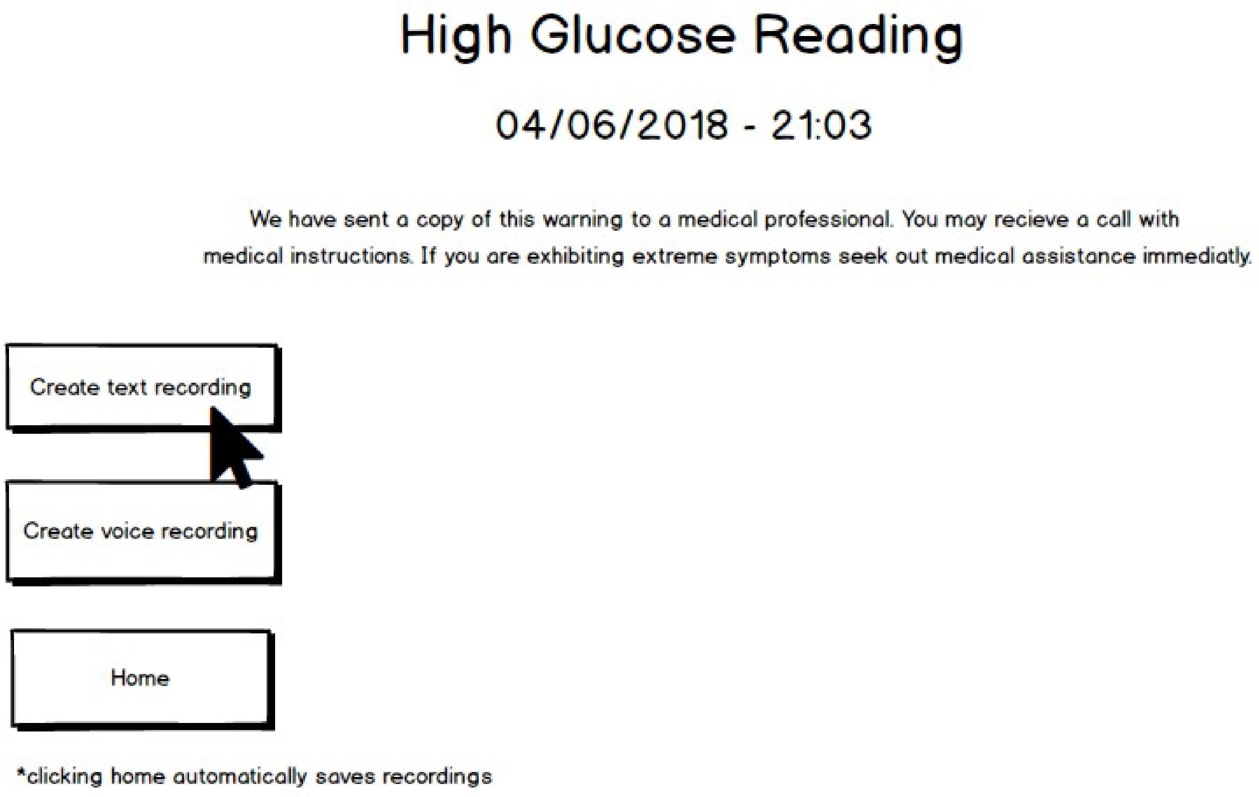


<<device>> PC, Tablet, Smartphone
<<OS>> Linux, Windows, Mac OS
<<component>> Browser
<<protocol>> HTTPS
<<device>> HTTP server
<<web server OS>> Apache
<<artifact>> existingPacificHealthServiceWebsite.js
<<device>> RTGMD
<<protocol>> Bluetooth
<<device>> smartphone
<<OS>> Google Android
<<artifact>> glucosMonitorApp.jar
<<protocol>> HTTPS
<<device>> HTTP server
<< web server OS>> Apache
<<artifact>> smartphoneInterface.jar
<<artifact>> frontLineInterface.jar
<<protocol>> AJP connector
<<protocol>> AJP connector
<<protocol>> HTTPS
<<device>> application server
<<application server OS>> Tomcat
<<artifact>> fetchData.jar
<<artifact>> saveData.jar
<<artifact>> checkGlucoseLevel.jar
<<device>> PC
<<OS>> Windows
<<artifact>> frontLineMonitorApplication.jar
<<protocol>> SQL
<<device>> Database server
<<DBMS>> mySQL Server
<<schema>> patient
Powered By Visual Paradigm Community Edition

Question 2: User Interface Design

Dialogues and Storyboards



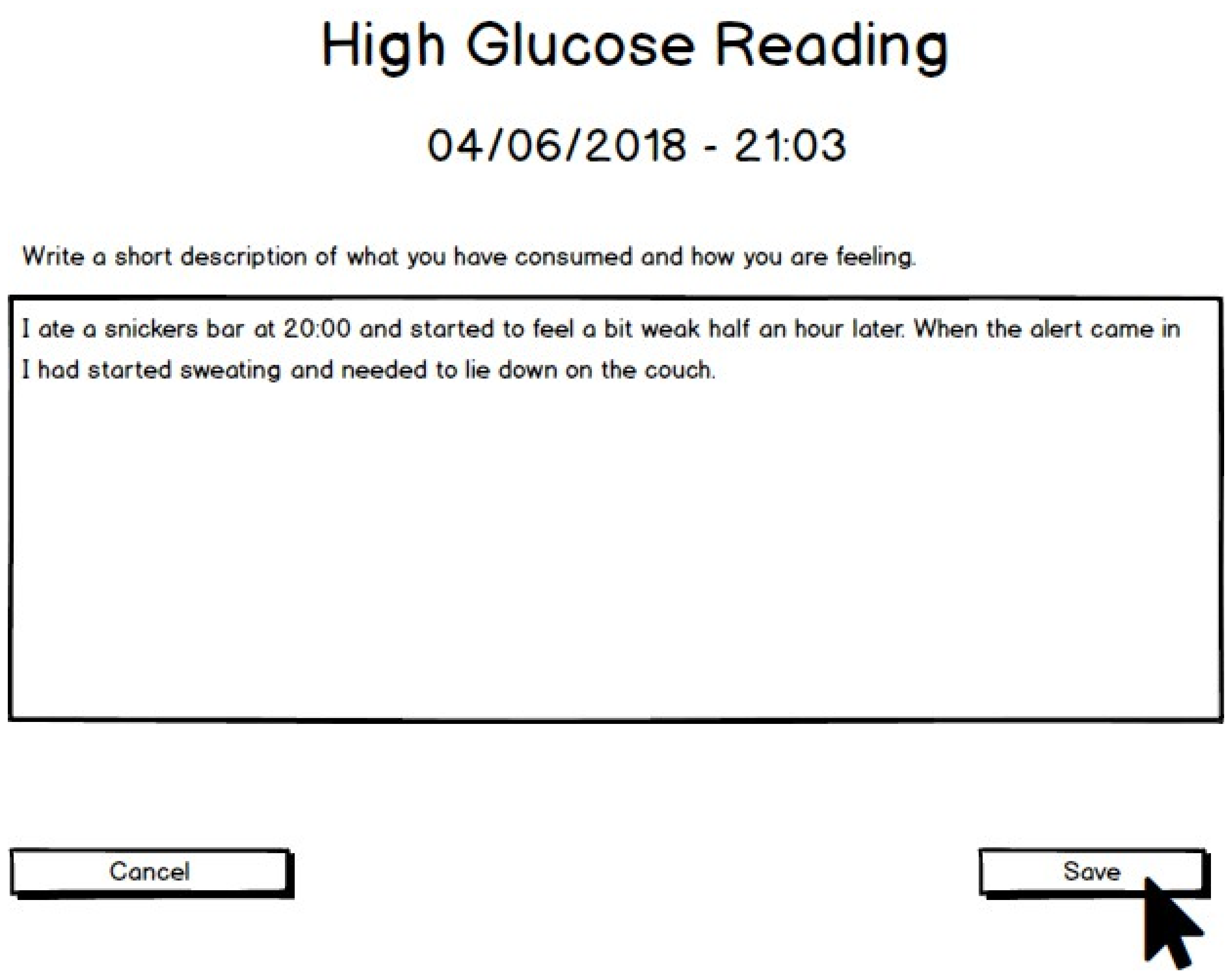


High Glucose Reading

04/06/2018 - 21:03

We have sent a copy of this warning to a medical professional. You may recieve a call with medical instructions. If you are exhibiting extreme symptoms seek out medical assistance immediatly.

*clicking home automatically saves recordings

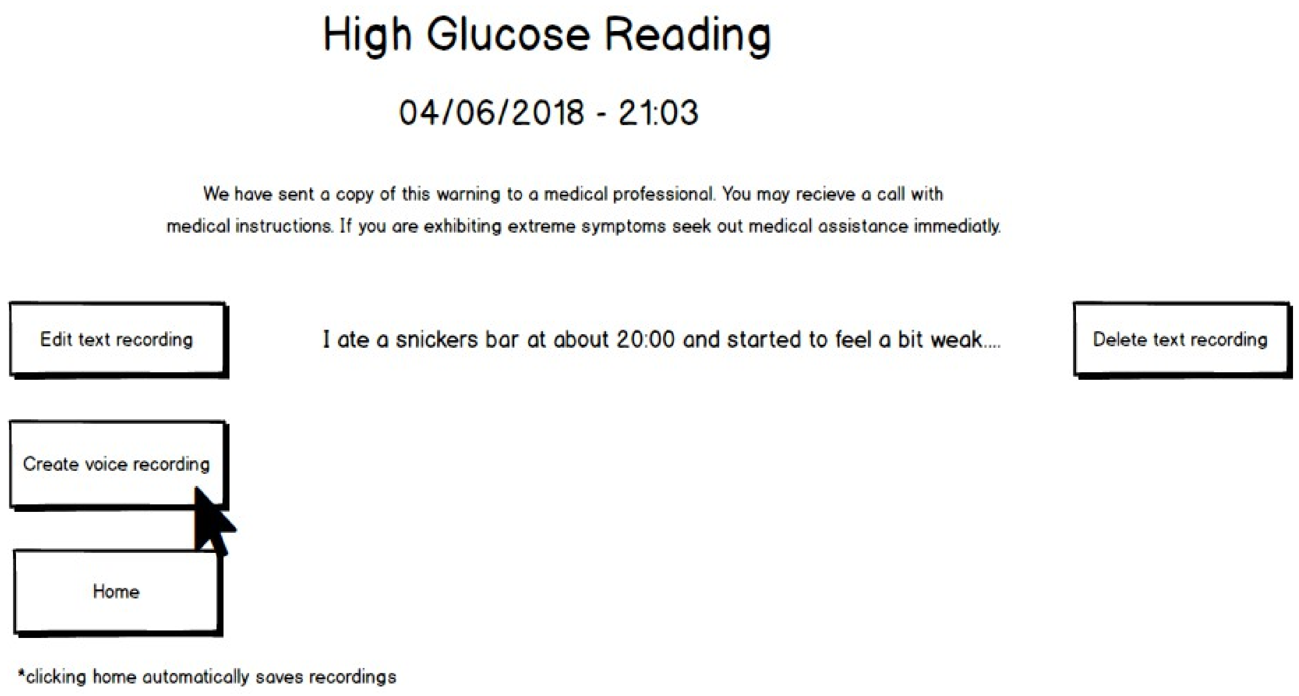


High Glucose Reading

04/06/2018 - 21:03

Write a short description of what you have consumed and how you are feeling.

I ate a snickers bar at 20:00 and started to feel a bit weak half an hour later. When the alert came in I had started sweating and needed to lie down on the couch.



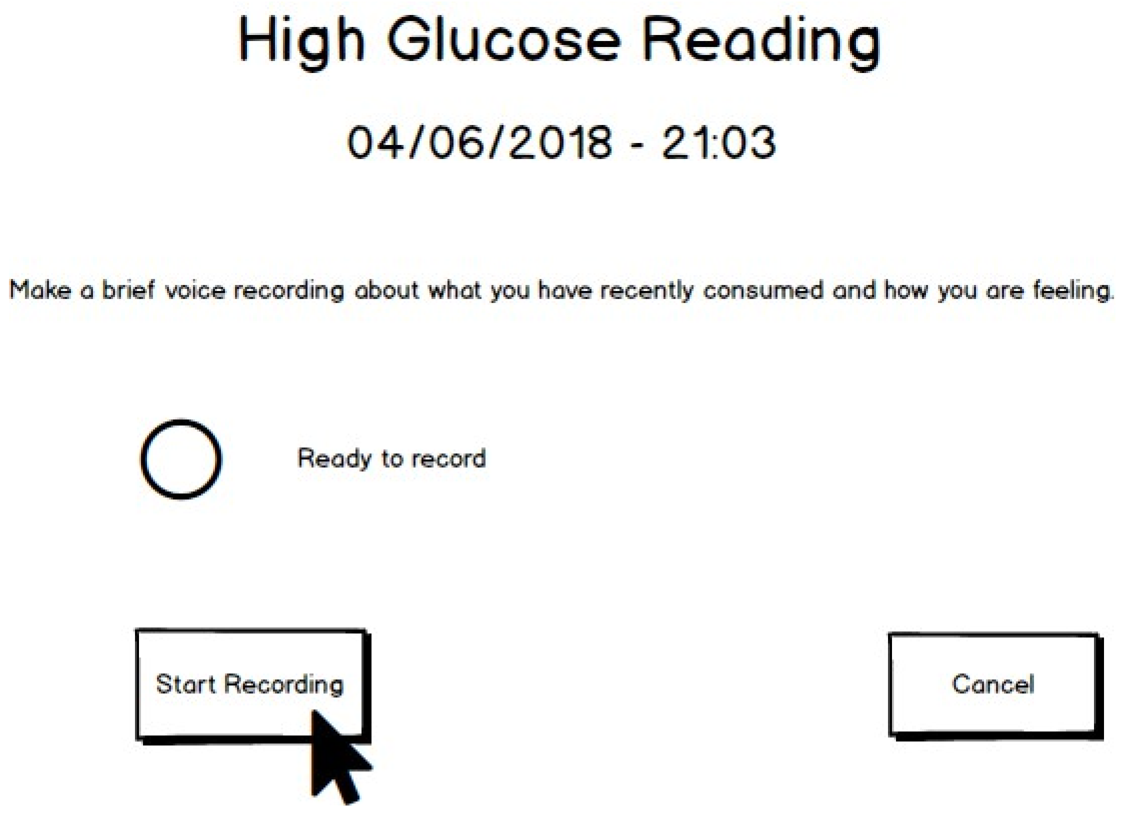
High Glucose Reading

04/06/2018 - 21:03

We have sent a copy of this warning to a medical professional. You may recieve a call with medical instructions. If you are exhibiting extreme symptoms seek out medical assistance immediatly.

I ate a snickers bar at about 20:00 and started to feel a bit weak....

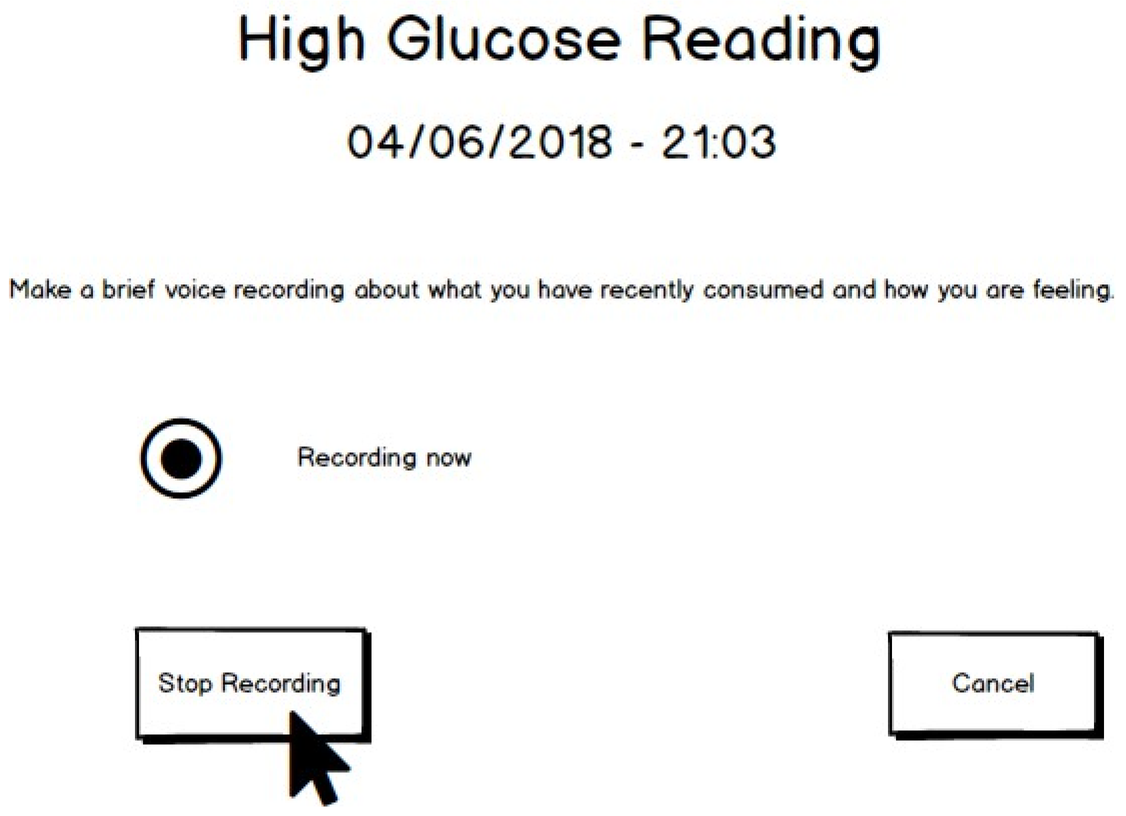
*clicking home automatically saves recordings


High Glucose Reading

04/06/2018 - 21:03

Make a brief voice recording about what you have recently consumed and how you are feeling.

Ready to record

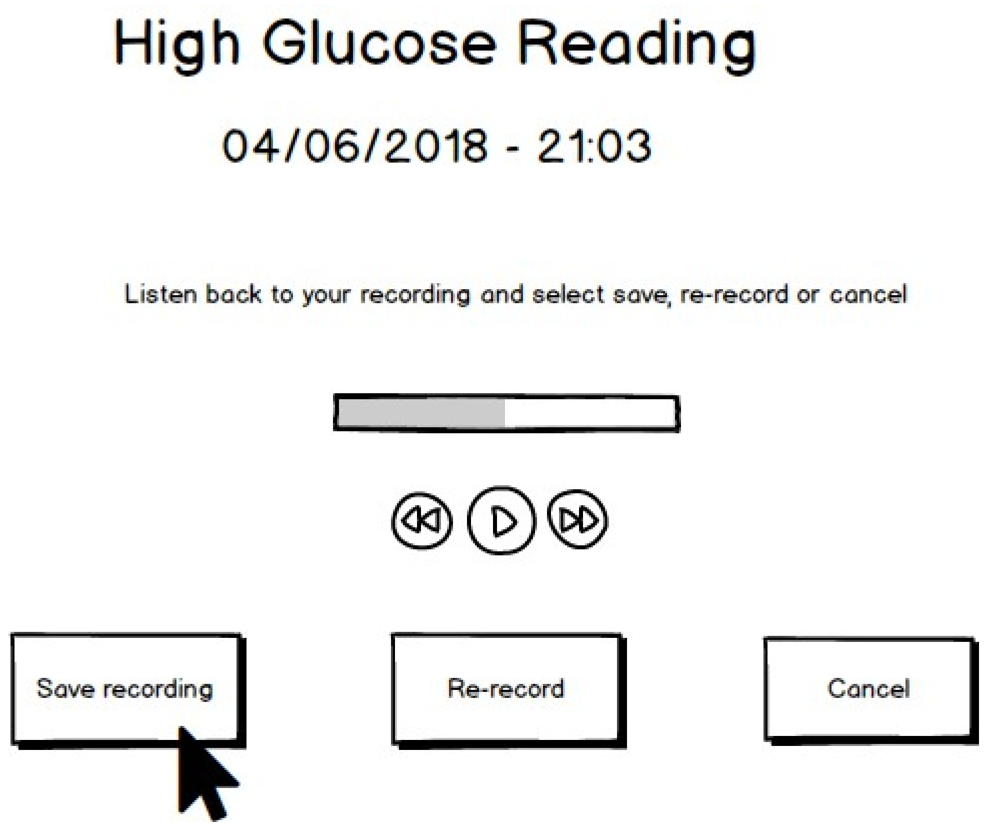


High Glucose Reading

04/06/2018 - 21:03

Make a brief voice recording about what you have recently consumed and how you are feeling.

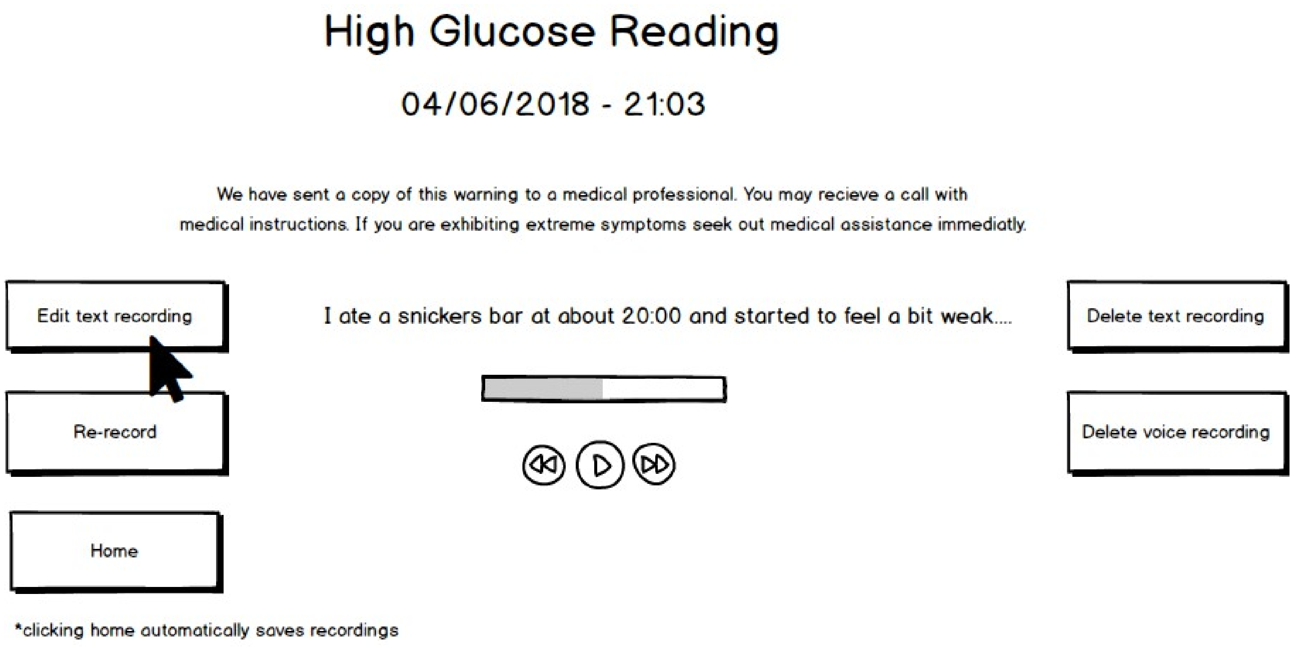
● Recording now



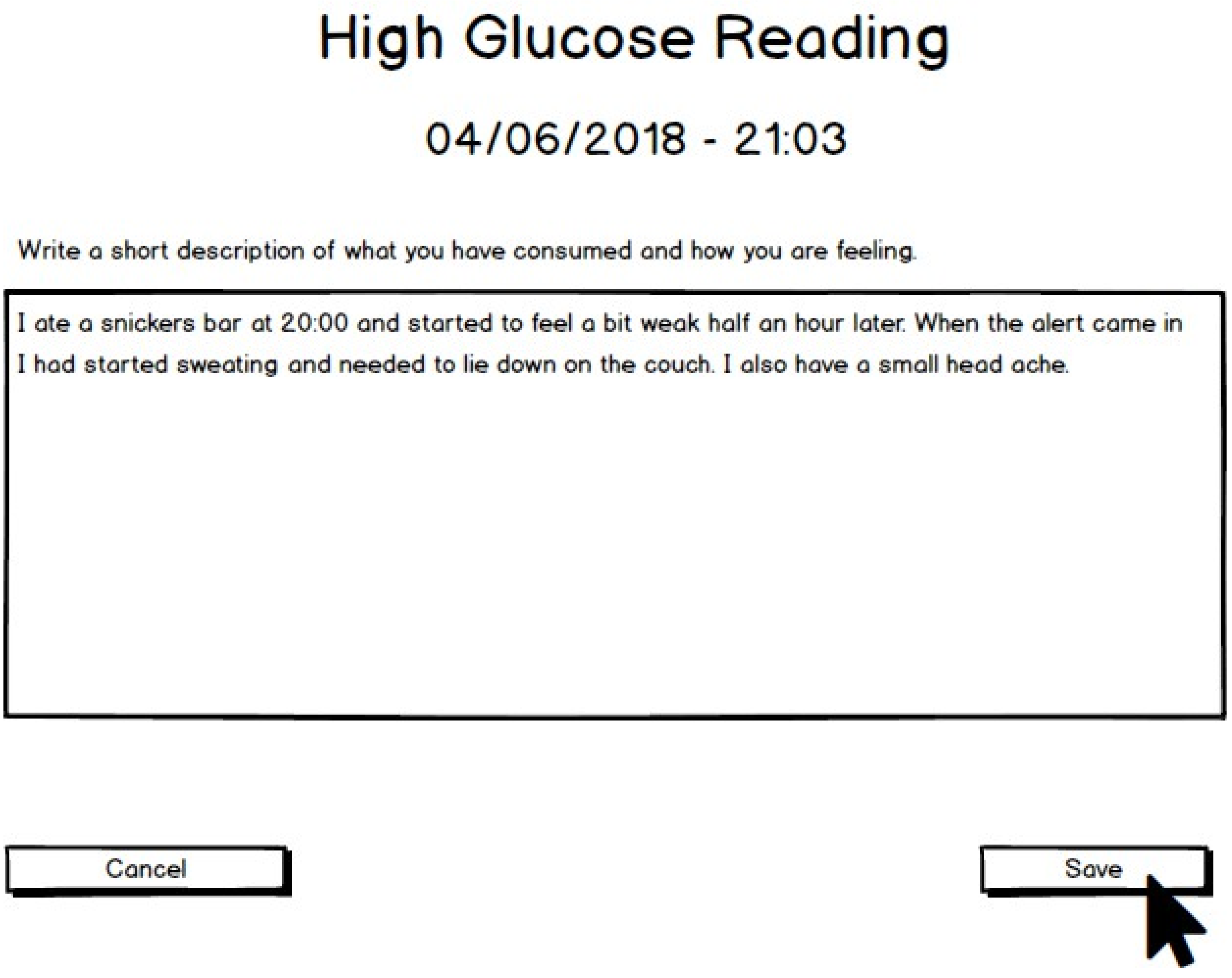
High Glucose Reading

04/06/2018 - 21:03

Listen back to your recording and select save, re-record or cancel



High Glucose Reading
04/06/2018 - 21:03
We have sent a copy of this warning to a medical professional. You may recieve a call with medical instructions. If you are exhibiting extreme symptoms seek out medical assistance immediatly.
Edit text recording
I ate a snickers bar at about 20:00 and started to feel a bit weak....
Delete text recording
Re-record
Delete voice recording
Home
*clicking home automatically saves recordings

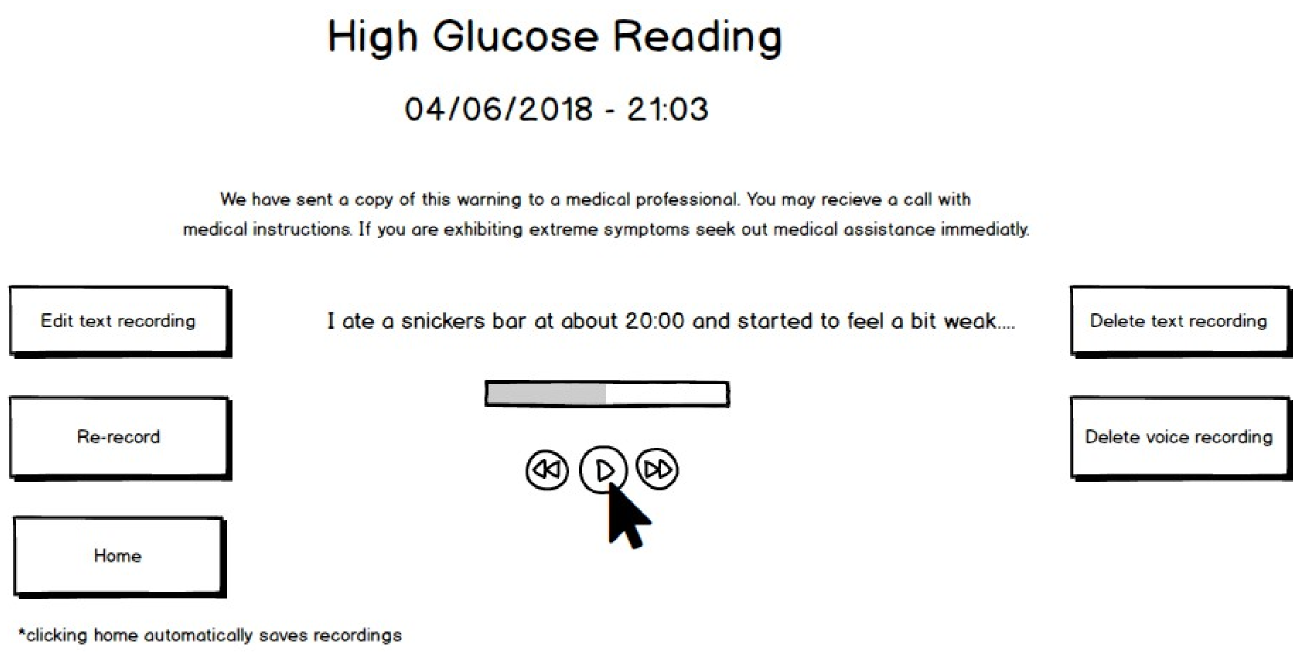


High Glucose Reading

04/06/2018 - 21:03

Write a short description of what you have consumed and how you are feeling.

I ate a snickers bar at 20:00 and started to feel a bit weak half an hour later. When the alert came in I had started sweating and needed to lie down on the couch. I also have a small head ache.



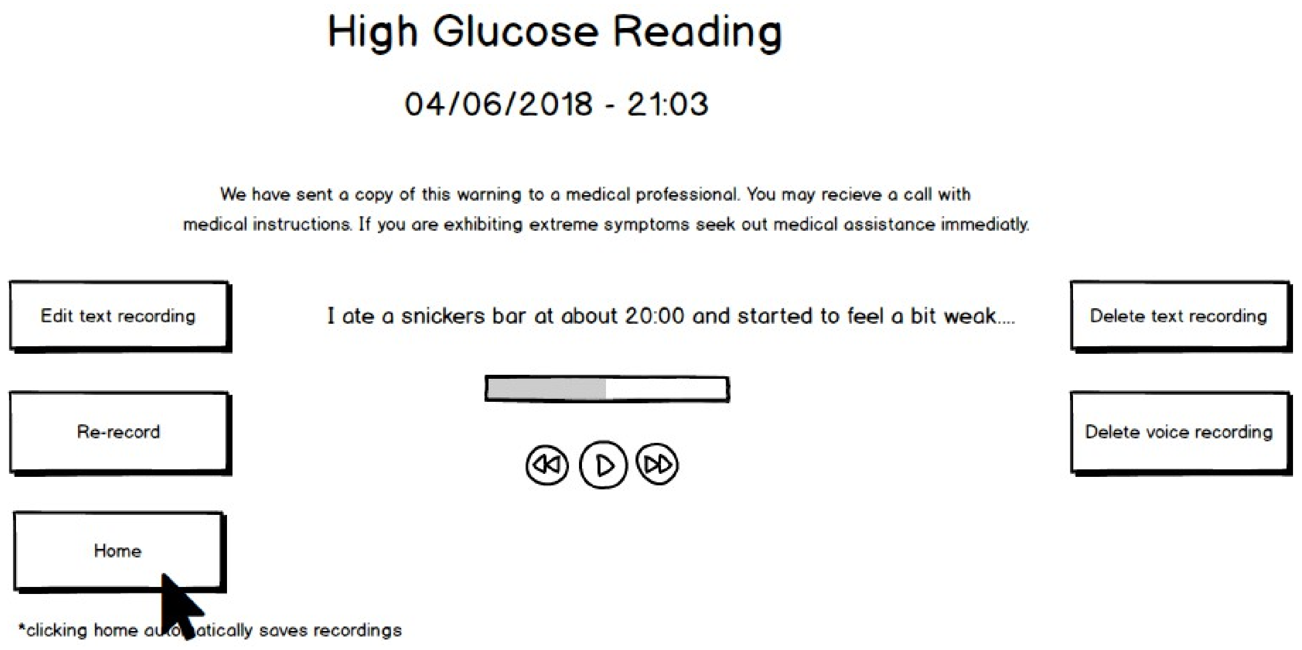
High Glucose Reading

04/06/2018 - 21:03

We have sent a copy of this warning to a medical professional. You may recieve a call with medical instructions. If you are exhibiting extreme symptoms seek out medical assistance immediatly.

I ate a snickers bar at about 20:00 and started to feel a bit weak....

*clicking home automatically saves recordings



High Glucose Reading

04/06/2018 - 21:03

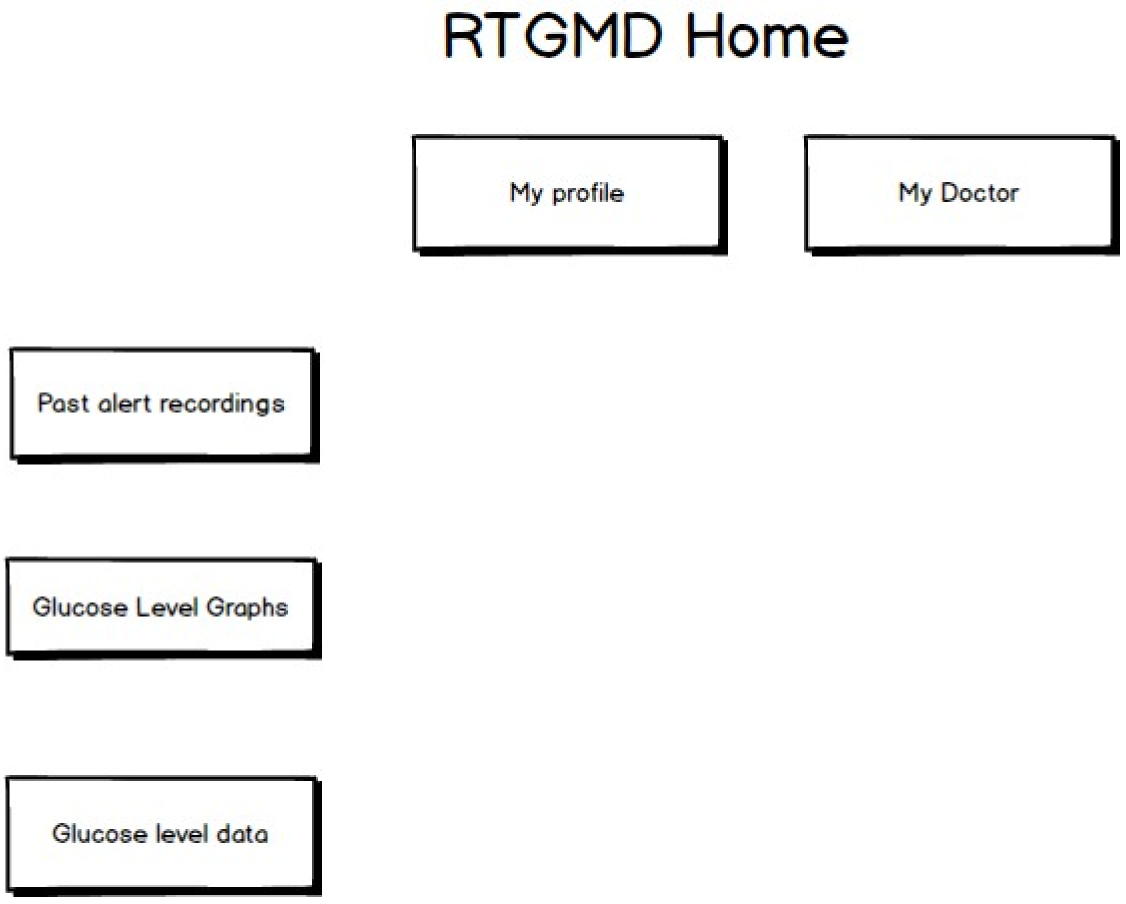
We have sent a copy of this warning to a medical professional. You may recieve a call with medical instructions. If you are exhibiting extreme symptoms seek out medical assistance immediatly.

I ate a snickers bar at about 20:00 and started to feel a bit weak....

*clicking home automatically saves recordings

Saving your recordings to profile...

Recording saved!


RTGMD Home
My profile
My Doctor
Past alert recordings
Glucose Level Graphs
Glucose level data

UI Design Principals

The alarm that sounds to warn the user of a glucose event will resemble that of an alarm to wake up. Almost everyone is familiar with alarms and the majority of people now use their smartphone as an alarm. This continuity will allow the user to instantly recognise the meaning of the sound being emitted from the phone.

In addition the 'alarm screen' that appears as the user looks at the beeping phone uses a large picture of a bell to re-enforce to alert. The bell is an ancient symbol synonymous with warning. A 'slide to acknowledge' widget will allow the user to suspend the alarm on the phone . The 'slide to acknowledge' widget is almost ubiquitous in mobile technology and is a commonly used for the function of 'waking up' mobile and tablet devices in hibernation mode. This means we can expect that the majority of users will be able use this function without thinking.

The 'alert sequence home' screen will only contain the time/date of the alert, the problem (high glucose, low glucose etc) and buttons 'Write text', 'Record Voice Message' and 'Cancel'. The screen only displays what is absolutely necessary for the user to fulfil the use case, in keeping with minimalist design principals. When a text is recorded the 'home screen' will be modified to display the beginning of the text that the user has written. This allows the user to see that the system has saved their words. The use of the users real words is more reassuring than a generic indicator stating that the message has been saved.

Voice recordings will be accompanied by the 'red circle' record symbol. This widely recognised symbol lets the user know that the recording function is now active, keeping the user aware of current system status. Once a voice recording has been made a 'play audio' widget will be placed on the 'alert sequence home screen' which allows the user to hear back their voice recording. Again the single triangle is widely synonymous with 'play'. This decision is in keeping with the UI principal of using well understood metaphors to communicate information quickly.

A home button is available on every screen in this application except the initial 'alert screen' which bring the user to the 'application home' screen. This allows users who do not wish to make recording to exit this part application quickly. I opted for the word 'home' over 'cancel' because cancel has connotations of permanent destruction and home of safe exploration. This is in keeping with the fact that the user can always come back and edit their recordings from main screen of the application. This contributes to making the system more flexible and encourages exploration.

Underneath the home button a small line of text informs the user that going home will automatically save recordings. This lets the user know that by pressing home they will not loose their recordings which keeps the user informed of the system status.

All screens in the application will have the same template. At the top of each screen the time/date of alert will be displayed along with the problem type (high glucose, low glucose etc). On each screen the 'Write Text', 'Record Voice Message' and 'Cancel' buttons are displayed in a column menu on the left hand side of the page. Displaying content on the left hand side is better for users from countries that read from left to right as our natural search for content starts on the left. Being consistent with the layout of each screen makes the display look more professional. It also contributes to efficiency of use as the experienced user will always know where to look for the feature they want to access.

When the user navigates away from the main screen they will always have the option to go back. If a user doesn't like a recording they can delete it and make a new recording. Once the user has saved or cancelled an instance of an alert, they will be able to search for and open that alert again if they want to edit their content. These are all features that aim to make any mistake easy to correct.

Once the user has finished with recording and clicked the 'home' button a message “Saving your recording to profile” will be displayed on the screen. This is followed by a screen with the message “Recording saved!” which times out after 4 seconds. This confirms for the user that their input has been stored. The user will then be redirected to the 'application home' page of the app.

Question 3: Database Design

PATIENT
patientID, lastName, firstName, medicareNumber, dateOfBirth, gender, race, height, weight, lowAlertLevel, highAlertLevel, drID, privateRecording
FOREIGN KEY (drID) REFERENCES Dr

DR
drID, lastName, firstName

FRONTLINE STAFF
frontlineMedicalID, firstName, surname

ACCESS LOG
access ID, patientID, drID, accessTime, accessDate
FOREIGN KEY (patientID) REFERENCES Patient
FOREIGN KEY (drID) REFERENCES DR

CHANGE LOG
changeID, accessID, oldLowAlertLevel, newLowAlertLevel, oldHighAlertLevel, newHighAlertLevel
FOREIGN KEY (accessID) REFERENCES Access Log

SMARTPHONE
smartPhoneID, phoneNumber, operatingSystem, osVersion, patientID
FOREIGN KEY (patientID) REFERENCES Patient

RTGMD
serialNumber, manufacturer, dateOfManufacture, firmwareVersion, smartPhoneID
FOREIGN KEY (smartPhoneID) REFERENCES Smartphone

NORMAL OBSERVATION
timeOfObservation, dateOfObservation serialNumber, glucoseLevel
FOREIGN KEY (serialNumber) REFERENCES RTGMD

ALERT OBSERVATION
timeOfObservation, dateOfObervation serialNumber, glucoseLevel, alertType, textResponse, voiceRecording. emergency, frontlineMedicalID
FOREIGN KEY (serialNumber) REFERENCES RTGMD
FOREIGN KEY (frontlineMedicalID) REFERENCES Frontline Staff

Table changes

- I have slit up person into doctor and patient as they are distinct objects with different roles in the system and they do not have the same relationships with other entities. For instance a doctor is not linked with a smartphone.

- Observation was split into normal observation and alert observation to avoid the many empty fields that would accumulate for the observations that do not have an alert. There is more information need about an alert observation and therefore more attributes needed in an alert observation table.
- I have removed the entities 'Alert', 'Text Response' and 'Voice Response' and made their attributes into attributes of the alert observation entity. A one to one relationships usually indicate that one table is really an attribute of another.
- Added Access Log table to record each access to the database and the doctor that made that access.
- Added Change Log table. This will monitor the changes made by the doctor to the patients alert levels.

Attribute changes

- I have added a lowAlertLevel and a highAlertLevel attribute to the Patient table. This will be the level at which a reading will trigger an alarm. This level can be set by the patients doctor.
- I have not included doctorID in the Smartphone table because in my implementation a doctor access the system via a website and is therefore not linked to a smartphone.
- I removed the ID attribute form the RTGMD as the serial number is a unique identifier that can be used as a primary key.
- I have added a boolean privateRecording (Y/N) field to the patient table which will allow the patient to dictate weather or not their personal written and voice recordings can be viewed by their doctor.
- I have added a boolean emergency (Y/N) field to the Alert Observation table to record weather or not frontline staff contacted the patient in the event of an alert. The frontlineMedicalID of the staff member that handled the alert is recorded weather or not the staff member contacts the patient.
- I have added dateOfObservation field to the observation tables. This will allow us to put the data in chronological order as we ORDER by date then ORDER by time. This will allow us to make our graphs.
- I have removed observation ID from the two observation tables (Alert Observation and Normal Observation) and used the time, date and serial number as the primary key for these tables.

Question 4: Role of Design Activities in the SDLC

The activities involved in designing a system in an iterative software development process take place in an order that allows each activity to provide the foundation for the next activity but leaves the door open for changes in the previous activity as unseen aspects of the problem are discovered over time.

The careful gathering of and prioritising of functional and non-functional requirements allows the designer to draft detailed system wide requirements specifications and software architecture notebooks, which help us define software architecture. At the same time the construction of these documents often uncovers additional requirements.

Prioritizing the non functional requirements is crucial for coming up with goals and philosophy of the project. When we have our functional requirements we know what we need to implement. We then must ask ourselves how do we implement? We can use a template to build a list of NFR's. As we build the list we must try to get a feel for the priorities. When we finish building our NFR list we must decide on which FURPS+ categories stand out. As a rule of thumb we should select two or three that we consider the most important. Most of the decisions will be made in reference to the top two or three priorities. The prioritised requirements can then be expanded upon as goals and philosophy which we can use to justify our decisions.

Another process involved in defining a software architecture is listing the assumptions, constraints and dependencies. It is important to be aware of the pre-existing conditions that will effect our system so that we can work with/around them. We should strive to identify as many of the assumptions and dependencies as soon as possible so we can consider them when we make decisions.

An important process is identifying the architecturally significant requirements. These are functional and non functional requirements that will effect how we build our system. These requirements that can be pointed to when me make an architecture decision. Each of these requirements should be satisfied by an architecture decision. Again it is important to try and list as many of these as possible as early as possible to help us make decisions.

Making architectural decisions is the final process in defining a software architecture. This is the point in which we identify devices, protocols and software that we will use to implement our system. It is important to justify decisions by listing the requirements that they satisfy. This will allow developers to see why you made your decisions. This is the final process because we need a good understanding of architecturally significant requirements, assumptions and dependencies to make decisions that best suit the system.

Once we have made out decisions we can model our system with a deployment diagram which can help us visualise the system.

At this point we should have noticed some hitherto unseen requirements. It is important to take those requirements back to the beginning, note them in our NFR table, update our assumptions and architecturally significant requirements and then alter/add decisions to satisfy these requirements. This iteration may reveal a few more requirements so we go through the loop once more. Ideally we would continue this process until all requirements are found and documented.

Usability requirements, use cases and heuristic guidelines help guide designers as they build and explain user interfaces which are then tested to reveal more usability requirements.

Systems analysis activities such as developing use case descriptions, activity diagrams and system sequence diagrams provide the foundation of our user interface design. Use case descriptions inform activity diagrams and system sequence diagrams which tell us how the user interface should be navigable by the user. We use activity and system sequence diagrams to build mockups. A mockup is a model that gives a skeleton view of the interface. As we build our mockup we should use heuristic guidelines to influence our design choices. The next process is testing where we make observations as either an expert or ordinary user to goes through our design. The observations gathered in the testing phase give us things to consider as we evaluate the design. After an evaluation we may need to go back to the design to make adjustments. This begins the design-prototye-test-evaluate cycle once again.

The functional requirements gathering that allows analysts to build entity relationship diagrams and domain model diagrams provide much of the work for the database designer, but similarly there are aspects of the problem that will come to light during database design that cause changes in earlier models.

The first process in building ERD's and domain class diagrams is discovering the entities/classes. This is done in the in the analysis phase by activities like the noun technique where documentation is searched through and the nouns listed. The list of nouns is then trimmed to contain only those nouns that the system must remember details about. This is our list of entities. Once the entities have been identified their attributes can be listed. Attributes are the details of an entity that the system must store information about. Many of these attributes will become the attributes of our database. We can then put the entities into ERD's/domain models which state their connections with other entities. These connections will tell us where we will need to put foreign keys in our database tables.

If building a relational database with the aid of ERD/domain models we can start by listing the entity names as tables and the entity attributes as database attributes. We must make sure each table has a primary key, that is one attribute that will reveal one instance of a table entry. At this stage the database is said to be in first normal form. We can then assign foreign keys. This is done by taking the primary key of an entity on the one side of a one to many connection and placing it into the many side table as an attribute. The use of foreign keys links the tables in our schema and allows us to navigate and find attributes. If we have composite primary keys we must ensure that none of the attributes are dependant on only one of the keys. If they are we need to make another table to remedy this. Our database will then be in second normal form. We then put our database into third normal form by ensuring all attributes functionally depend only on the primary key. Attributes that are functionally dependant on other keys can be placed into a new table.

Once the database is in third normal form we can check things like 1-1 relationships. 1-1 relationships may be suitable for ERD/domain models but in a database they can indicate that one table may be better placed as an attribute of the table it its connected to. At the end of all these processes we have an effective relational database that should not contain any redundant data.

The database tables and attributes may be effected by new requirements. It is important to consider the database when updating the software architecture documentation to record new requirements.