Description

Welcome to DCA (Diabetes Control & Automation) Wiki. In this documentation you will find the details about this project and how we have been working, and unlike README, we will explain more in depth the motivations of doing this project.

Alexis Mengual & David Usón
proyectocadams@gmail.com

About DCA	Project Details	Technologies used
		
About DCA	Project Details	Technologies used
Introducing DCA - why we built it and what it does.	Documentation about how we have been working - backlogs, sprints…	Explanation about what languages and frameworks we use and why we have considered using them.

What is DCA?

DCA (Diabetes Control & Automation) is a project who seeks to help, using the new technologies, managment the diabetes in the most automated way posible.

How the idea came about?

In the current era, we can see how the new technologies bring us more and more, facilities for our daily life, how for example, we can have a smart house thanks to IA like Alexa or Google Home, also we have any information in any place thanks to smartphones, but as for chronic diseases, which can also be part of the daily life, the facilities are not so many and are more limited.

And in this project, we focus

What kind of languages we are use?

In this project, we thought that a language that brought us something different from what we saw during our AMS course and that was better suited for our work was React Native next to the Expo framework.

Why React Native?

React Native is a language created by Facebook in 2015. This language allow us programming in JavaScript native and create applications mobile for Android and iOS. So, why we used React Native and not Java? How we said before, React Native allow us create an application for Android and iOS, but another advantage against Java is that it processes information much faster than an application made with it. Also exists anothers disadvantage against Java, but also is true that we wanted to contribute something different.

Also, we thought about using Angular, but the learning curve was a little more difficult compared to React, and it's not that fast.

Who is Expo Framework?

Expo is a framework that allow us a create an application web using React Native, also he has another libraries to facilitate some tasks.

This converts the React Native code into ReactJS and in this way, having a Web Application identifies the content of the mobile application.

And finally, React Nativo can work with Expo SDK and we can develop the app and emulate it at the same time in a real mobile without having to emulate it with Android Studio or other alternatives, which gives much more comfort to test our components and faster.

As a matter of fact, Expo allows you to mount the application and publish it in the App Store and Play Store, in our case we will not do so because we consider that it is not yet a large enough application to be used by other users.

What about Jest?

Jest is a framework that allows us to test our application components. We use it to be able to mock the entire part related to NFC data collection and save from it.

For those who do not know the concept of Mocking an application, mocking serves the following points:

- It allows us to simulate certain aspects of the application with false data, for example, to access a remote resource and obtain information from it.
- It allows us to prepare and establish all the criteria at the business level of the application so that, when we have access to those resources, the implementation is much faster and more secure.

Mocking the application for the above mentioned case has been very useful to us, since due to the inconvenience about the

COVID-19 we could not have access to anything that could prove that our application supports NFC readings.

What about Firebase?

Firebase is a platform created by Google, whose main function is to develop and facilitate the creation of quality apps with the tools it offers. Tools such as analytics, authentication, databases…

In this project we have not entered into all the possibilities offered by Firebase, in certain cases because we were not interested or did not have time to make good use of them, below we will comment on those that if we have used:

- Authentication: It allows us to manage all types of authentication and synchronize them with Firebase, from registering with a user with e-mail and password, to accounts already registered with Google Account or Facebook, even to do so by means of a fingerprint reader on mobile phones with this option.

In our case, the main idea was that our application would work with user registration by e-mail and password and Google Account, but because of a compatibility problem and we did not find the solution, we changed it for authentication with Facebook.

- Firestore Cloud: Firestore Cloud uses a Nosql structure, very similar to the one studied as Mongodb. Firestore also offers us the use of Realtime Database, so all data is synchronized on mobile devices.

- Cloud Functions: This tool is one of the most useful tools offered by Firebase, this tool allows to execute backend code in response to events that are activated by HTTPS requests. This tool can be very useful for the topic of updates, this is used a lot in mobile applications, as Whatsapp when creating backups of our conversations, but for a while, we could not find a utility and could notimplement it.

Project Details

In this document we will show the details of how we have worked, the issues we have dealt with and the restructuring.

Flowchart

The App is used to connect to the NFC device with Firestore Cloud. This, scans the device and sends the data to the Firestore Cloud at the same time that it shows them in it.

Goals

- [x] Multilanguage Module

  Get the application to support different languages, depending on the language the user has indicated on their operating system.

- Available languages:

  - es_ES
  - en_US
  - pt_PT

- [x] Login with *Email/Password* Module

  In this module the application will allow create account with email and password.

- [ ] Login with Google Account (Cancelled)

  In this module the application will allow sign in with Gmail Account.

- [ ] NFC Reader Module.

  This module will get the information about the glucose from the glucometer.

- [ ] Module about the Information about the Pacient.

  It will have the functionality of generating reports on how the patient manages his diabetes. So it will allow you to send information or to be able to correct errors more accurately and easily.

Backlog

In this page you will find the all project specifications.

- [x] Make a Architecture Project sprint1
- [x] Configure Firebase Authentication sprint1
- x] Make LoginScreen sprint1
- [x] Make Authentication Function (with Email / Password) sprint1
- [ ] Make Authentication Function (with Google Account) sprint1 cancelled
- [x] Make Authentication Function (with Facebook) sprint1 hotfix
- [x] Unit Test about Authentication sprint1
- [x] Integration Test about Authentication sprint1
- [x] Make ReaderScreen sprint1
- [x] Multilanguage sprint1
- [x] Configure Firestore Cloud sprint2
- [x] Fix Authentication Functions sprint2 hotfix
- [x] Make UserScreen sprint2
- [x] Make Medication Configure Functions sprint2
- [x] Add Historial Navigation in BottomTab Navigation sprint2
- [x] Make TextInput Component for Settings in User Screen sprint2
- [x] Make Historial Screen sprint3
- [x] Create a Collection in Firestore of user data collected per day sprint3
- [x] Make a DatePicker Modal to search user resume per day sprint3
- [x] Calculate Hba1c and eAG per day sprint3
- [x] Calculate Hba1c and eAG per month sprint3
- [x] Calculate Hba1c and eAG per trimester sprint3
- [x] Make a report with the resume of user data sprint3
- [ ] Generate a PDFReport sprint3 cancelled
- [x] Generate a Email with the Report sprint3 hotfix
- [ ] Apply SOLID Principles (as far as possible) sprint4
- [ ] Make mock for simulate a 'Reader Process' sprint4
- [ ] Fixing errors and bugs sprint4
- [ ] Finish pending specifications sprint4

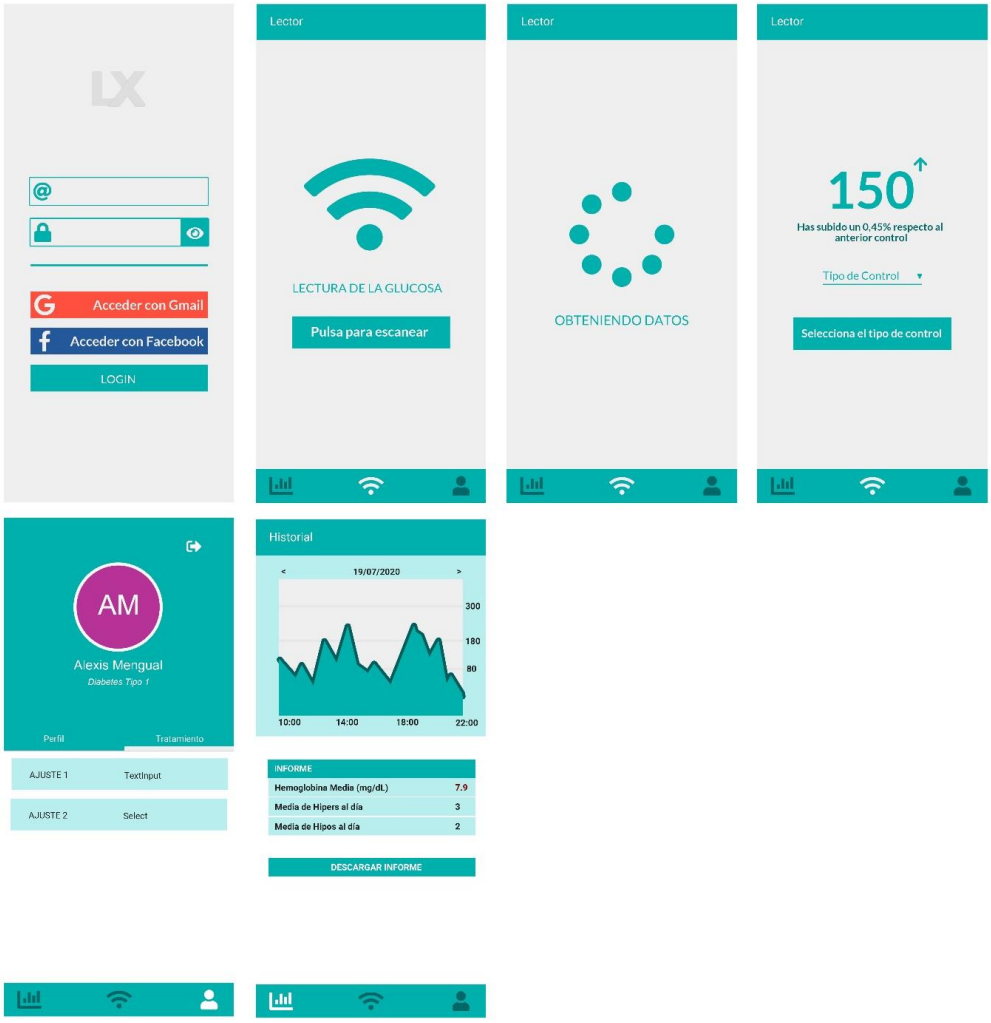
Sprint 1

Specifications

In this sprint we maked the next funcionalities:

- Project Structure.
- Installation of necesary libraries.
- Authentication configure with Firebase.
- Multilanguage configure.
- Screens:
    - Login Screen
    - Home Screen
    - Reader Screen
    - *Navigation Menu*

Wireframes



Incidences

During Sprint 1, we had certain incidents that have caused us to have to look for other ways and restructure certain aspects of the application so that it does not in any way affect to a large extent the objectives set:

- Google has not compatibility with Expo CLI and not permit do login with a Gmail user. We decide replace Google Login for Facebook Login

Sprint 2

Specifications

In this sprint we maked the next funcionalities:

- Configure Firestore Cloud for make a CRUD
- Installation of necesary libraries.
- Register with Firebase (Email/Password).
- Make form to configure the medication instruccions for user.
- Include in bottom tab, the "Historial Navigation".
- Screens:
    - Register Screen
    - User Screen

Fixes

During sprint 2, we modified and fixed certain aspects of the previous sprint. * The functions for Sign In in our project, we move to another files inside the functions folder, we have always tried to comply with the S of SOLID, but always looking first for the application to be functional before.

Incidences

During Sprint 2, we had certain incidents that have caused us to have to look for other ways and restructure certain aspects of the application so that it does not in any way affect to a large extent the objectives set:

- To add Firestore Cloud libraries to our project, we had to install it with expo install, not npm install.

Sprint 3

Specifications

In this sprint we maked the next funcionalities:

- Installation of necesary libraries (react-native-pdf-lib, react-native-modal-datetime-picker). (1 hora/s)
- Create a Collection in Firestore of user data collected per day. (2 hora/s)
- Make a DatePicker Modal to search user resume per day. (2 hora/s)
- Generate Chart with values from Collection. (3 hora/s)
- Calculate Hba1c and eAG per day / month / trimester (2 hora/s)
- Make a report with the resume of user data (4 hora/s)
- Generate a PDF Report (5 hora/s)
- Fixes (8 hora/s)
- Testing (6 hora/s)
- To Investigate about the React and Frameworks (4 hora/s)
- Screens: (5 hora/s)
    - History Screen
    - User Screen (Updated)
    - Reader Screen (Updated)

Fixes

During sprint 3, we modified and fixed certain aspects of the previous sprint.

- We created new componentes with a view to simplifying and separate the work of the screens.

Incidences

During Sprint 3, we had certain incidents that have caused us to have to look for other ways and restructure certain aspects of the application so that it does not in any way affect to a large extent the objectives set:

- The library to generate a PDF file, which we would use to generate the report, does not work properly. The alternative option is send a email with this report.

Sprint 4

Specifications

In this sprint we maked the next funcionalities:

Fixes

During sprint 4, we modified and fixed certain aspects of the previous sprint.

Incidences

During Sprint 4, we had certain incidents that have caused us to have to look for other ways and restructure certain aspects of the application so that it does not in any way affect to a large extent the objectives set: