DocTalk Documentation

Antonio Herrera

Daschel Cooper

Andrew Jensen

Kevin Nguyen

Henry Schliebe

Duc Phan

Table of Contents

Brief Description

The goal of DocTalk is to keep a establish line of communication between the doctor and the patient as it is important to discuss and assign medical attention such as keeping tabs on usage of medicines, creating appointments, and giving doctors the data for each patient to see progress of the patient's health whether it is getting better or worse and establish the next step to give to the patient in order to maintain the healthy balance that each patient needs.

The app helps doctors look at data for each patient and keep tabs for each patient to see how well each patient is handling their health after the first appointment. The app as a login for both the patient and doctor with a username and password, then the home page shows a calendar for both the patient and doctor with events and appointments that the doctor created for the patient and has the data of the patient such as pulses, vitals, temperature, and weight.

Tools

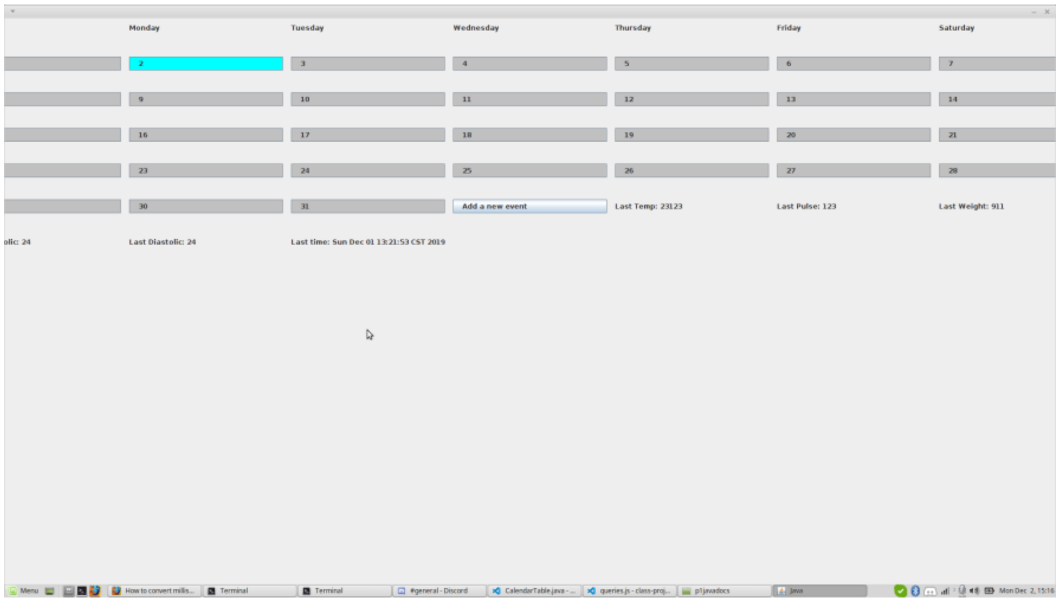
The frontend of the project is an application developed in Java, using JUnit for testing. Maven is used to make building the project easier.
Java: https://www.java.com/en/
Maven: https://maven.apache.org/
JUnit: https://junit.org/junit5/

The backend of the project is a database powered by Node.js and SQLite. The built-in testing functionalities are used.
Node.js: https://nodejs.org/en/
SQLite: https://www.sqlite.org/index.html

Architecture

Front end

Main front end consists of frames and utilities.
Frames are used to view/control data. For example, the login frame creates a window that lets the user puts in their login information, checks their input against the backend and lets them log in.
The utilities are classes that serve various functions, such as InputValidation that can be called into other frames and used.



	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
	2	3	4	5	6	7
	9	10	11	12	13	14
	16	17	18	19	20	21
	23	24	25	26	27	28
	30	31	Add a new event	Last Temp: 23123	Last Pulse: 123	Last Weight: 911
olic: 24	Last Diastolic: 24	Last time: Sun Dec 01 13:21:53 CST 2019				

- Example of the calendar frame

Back end

Main infrastructure is token based packets using json web tokens jwts that are sent to the backend. The backend parses them, and we have individual functions that are generalized so they can be used anywhere. The back end will build their own SQL requests. Gitall and Run. If the request is good, it will send out a web token of what the database is returning back to the front.

Appendix

- Building
  - FRONTEND
    - From the `‘healthapp’` directory, run `‘mvn clean install’` then `‘mvn test’` and `‘mvn exec:java’`
  - BACKEND
    - Run `‘node apilite.js’` and `‘sqlite3 sqlite.db’`

- Classes
  - Frames
    - CalendarTable
    - DailyLogFrame
    - EventFrame
    - HomeFrames
    - LoginFrame
    - RegisterFrame
    - ReminderFrame
  - Models
    - DoctorData
    - PatientData
    - Reminder
    - TestData
    - UserData
    - VitalsData
  - Utilities
    - Auth
    - InputValidation
    - Token