**UNIVERSITATEA DIN PITEȘTI**

**FACULTATEA DE ȘTIINȚE, EDUCAȚIE FIZICĂ ȘI INFORMATICĂ**

**PROGRAMUL DE STUDII UNIVERSITARE DE LICENȚĂ: INFORMATICĂ**

**LUCRARE DE LICENȚĂ**

**Coordonator științific,**

**Lector.univ.dr. Maria Miroiu**

**Absolvent,**

**Nicolescu Mihai-Robert**

**Pitești**

**2021**

**UNIVERSITATEA DIN PITEȘTI**

**FACULTATEA DE ȘTIINȚE, EDUCAȚIE FIZICĂ ȘI INFORMATICĂ**

**PROGRAMUL DE STUDII UNIVERSITARE DE LICENȚĂ: INFORMATICĂ**

**Sistem informatic pentru monitorizarea nivelului unui râu**

**Coordonator științific,**

**Lector.univ.dr. Maria Miroiu**

**Absolvent,**

**Nicolescu Mihai-Robert**

**Pitești**

**2021**

**Cuprins**

[**Introducere**](#_m0ik0l87uvz3) **3**

[**Capitolul I - Tehnologii Web**](#_xrq6ddjorzwz) **5**

[1.1 Vue.js](#_41awz0ywhwyy) 5

[1.1.1 Vue-CLI](#_9a7tubscqbre) 5

[1.1.2 Sistem reactiv de legare a datelor](#_e3txvlrw4q4x) 6

[1.1.3 Sintaxa sistemului de legare a datelor](#_u32bm0w0pnbb) 7

[1.1.4 Componentele din Vue.js](#_wp3nb66miqr0) 8

[1.1.5 Directive Vue](#_14npongcmxen) 10

[1.1.6 Rutarea componentelor](#_ekhhl91gas0l) 13

[1.2 Vuetify](#_68qkrhy1mbxz) 15

[1.2.1 Instalare](#_97ox1zh941sx) 15

[1.2.2 Tipuri de teme și configurarea acestora în Vuetify](#_9y1dj0ke4wm3) 15

[1.2.3 Componente Vuetify](#_bmjcohdn2nn4) 17

[1.3 JavaScript](#_663e5wclt949) 25

[1.3.1 Descriere](#_it6x9lmrlx5y) 25

[1.3.2 Vue și JavaScript](#_ifhpyk61gfuo) 26

[1.3.3 Calcularea datei](#_2f4c0qt59ql0) 28

[**Capitolul II - Tehnologii pentru colectare de date**](#_y266o3hqparp) **30**

[2.1 Raspberry PI](#_lotki9xjzq2c) 30

[2.1.1 Raspberry PI 4 Model B](#_xvcnp8dvykxi) 31

[2.1.2 Sistemul de operare Raspberry Pi OS](#_cywb7r5gkmfw) 34

[2.1.3 Instalarea sistemului Raspberry Pi OS](#_fn0wpd6w38bu) 35

[2.1.4 GPIO](#_d5zgv8xijfoa) 36

[2.2 Python](#_mr3ldrn46q9h) 39

[2.2.1 Avantaje ale limbajului Python](#_czxee0igu6cl) 41

[2.1.3 Module](#_cowlm5cia8a4) 42

[2.2.3.1 PiCamera](#_h304e0ethjyt) 43

[2.2.3.2 Time](#_43zgsggwiccf) 45

[2.2.3.3 Datetime](#_4em1yymyvw99) 46

[2.2.3.4 RPi.GPIO](#_k5mmh6tgsp35) 46

[2.2.3.5 base64](#_gr2dymev3jux) 48

[2.3 Google Cloud Platform](#_qdsczw59ubjw) 50

[2.3.1 Firebase](#_vqycszp8ar9c) 51

[2.3.1.1 Firestore](#_j1a63gtz2vq0) 54

[2.3.1.2 Hosting](#_9gpc1f4mhu13) 55

[2.3.1.3 Firebase CLI](#_781qc1scbo3s) 58

# 

# Introducere

Din cauza schimbărilor climatice, crește intensitatea și frecvența evenimentelor meteorologice extreme, precum temperaturile extreme, inundațiile și secetele. Evenimentele acestea rezultă deseori în pierderea vieților unor persoane sau animale și pierderea de proprietăți, în special în țările care nu au un mod de a preveni dezastrele naturale din cauza dificultăților financiare sau sociale. Inundațiile în special reprezintă una dintre cele mai periculoase situații de urgență pentru țara noastră, ele apar atunci cand albia unui râu nu poate prelua toată apa de la ploaie.

Termenul Smart city este unul care se aude des în zilele noastre, reprezinta o varietate de solutii IoT integrate in viața zilnică a orașului, în beneficiul locuitorilor săi. Prezentată în această lucrare este una dintre aceste soluții, un sistem de monitorizare a râurilor care facilitează detectarea din timp a inundatiilor. Această soluție constă într-un senzor cu ultrasunete montat la un Raspberry PI și o cameră care face poze la râul respectiv la un anumit interval de timp.

Aplicația realizată va fi folosită pentru a monitoriza nivelul unui râu, a colecta și afișa date citite prin senzor de la râul respectiv. Și, în caz că nivelul acestuia este unul anormal, va trimite mai departe o alertă care ar trebui să ajute la prevenirea unei posibile inundații în împrejurimile râului. Din fericire, de cele mai multe ori, durează mai multe ore sau chiar zile pana cand se creeaza o inundație. Astfel, autoritățile și locuitorii se pot pregăti din timp pentru acest eveniment.

Vizualizarea și editarea datelor se face folosind o aplicație WEB, realizată cu framework-ul Vue.js. În primul capitol, intitulat ,,Tehnologii Web” sunt prezentate informații generale despre tehnologiile de front-end și de back-end folosite pentru crearea aplicației WEB, felul în care tehnologiile comunica una cu cealaltă, și motivul pentru care acestea au fost alese în defavoarea altor tehnologii similare.

În cel de-al doilea capitol va fi descris modul în care funcționează colectarea de date. Aceasta se face folosind un senzor ultrasonic de distanță HC-SR04+, senzorul este activat de un script, scris în limbajul de programare Python care rulează pe un Raspberry Pi 4 Model B cu o memorie RAM de 2GB.

În ultimul capitol va fi prezentată aplicația în sine, cum au fost folosite tehnologiile menționate în capitolele anterioare și părți din cod în care se explică felul în care a fost construită aplicația, atât partea de front end cât și de back-end: adăugarea datelor și interogarea bazei de date Firestore de la Firebase, modul în care aplicația procesează datele respective pentru a detecta nivelul râului, și cum le afișează mai departe utilizatorilor.

# Capitolul I - Tehnologii Web

## 1.1 Vue.js

Vue este un framework progresiv folosit pentru construirea interfețelor interactive pentru utilizatori. Este unul dintre cele mai folosite framework-uri pentru simplificarea dezvoltării paginilor web. Biblioteca de baza folosita de Vue se concentreaza doar pe partea vizuală și este ușor de integrat cu alte biblioteci sau proiecte pentru o experiență completă. Astfel, Vue este conceput pentru a fi adoptat în mod incremental. De asemenea, poate să fie folosit pentru aplicații de tip single page, atunci cand este utilizat în combinație cu alte biblioteci moderne.

Caracteristici ale frameworkului Vue.js:

### 1.1.1 Vue-CLI

Vue.js poate fi instalat folosind prin command line folosind interfața vue-cli folosind comanda npm install -g @vue/cli.

Aceasta ajută la compilarea și construirea proiectului cu ușurință. CLI-ul este un pachet npm instalat global și oferă comanda vue în terminal. Folosind comenzile, exista posibilitatea de a crea un nou proiect simplu și rapid prin comanda vue create și numele proiectului. De asemenea, există și comanda vue ui care oferă o interfata grafica prin care se pot gestiona și crea proiecte noi.

Cand se creeaza un proiect prin Vue-CLI, acesta instalează un binar numit vue-cli-service care poate fi folosit prin comanda npm run serve.

Comanda vue-cli-service porneste un dev server bazat pe webpack-dev-server unde se poate vizualiza proiectul vue local. Una din caracteristicile importante ale acestuia este ca are funcția de auto-refresh: ori de câte ori se modifică fișierul .vue pe care se lucrează, pagina se reîmprospătează automat fără a fi nevoie de interacțiunea dezvoltatorului. Acest auto-refresh face parte din Hot-Module-Replacement, o funcționalitate care schimbă, scoate sau adaugă module în timp ce aplicația rulează fără a fi nevoie de o reîncărcare completă. Funcția de Hot-Module-Replacement poate îmbunătăți viteza dezvoltării unei aplicații prin:

* păstrarea stării aplicației (care normal s-ar pierde în timpul unei reîncărcari complete)
* actualizarea instantanee a browserului în momentul în care se fac modificări la CSS/JS în codul sursa; timpul de actualizare este comparabil cu schimbarea de CSS direct folosind instrumentele de dezvoltare ale browserului.

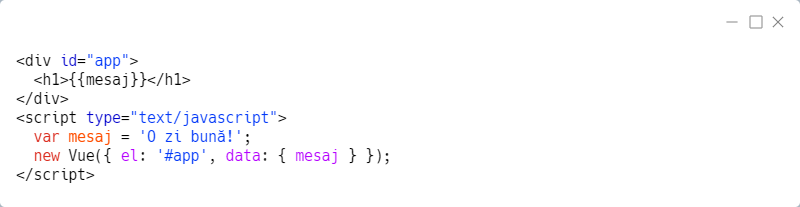
Unele din comenzile disponibile în Vue-CLI sunt:

* --open; porneste browserul atunci cand se porneste și serverul
* --copy; copiaza url-ul serverului în memorie cand porneste serverul
* --host; specifică host-ul dorit
* --port; specifică port-ul dorit
* --https; implicit este setat ca și false. Odată activat specifică serverului că trebuie să ruleze pe https.

### 1.1.2 Sistem reactiv de legare a datelor

Una dintre caracteristicile de bază ale framework-ului Vue este sistemul reactiv de legare a datelor. Vue pastreaza datele precum variabilele Javascript, vectorii, matricile sincronizate cu DOM-ul fără a fi nevoie de intervenția dezvoltatorului.

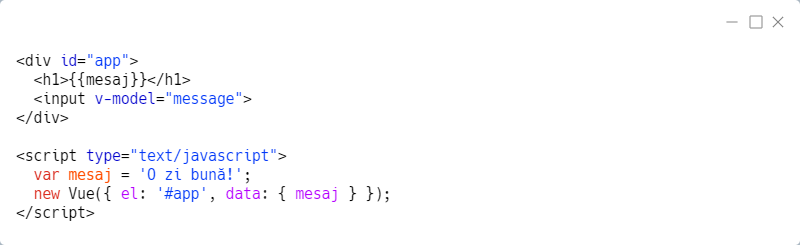
Exemplu:



În Vue, textul prezent în elementul h1 din exemplu are numele de expresie de legatura. Dacă valoarea variabilei mesaj va fi schimbată în javascript, se va schimba și valoarea variabilei din DOM.

Pe lângă sistemul reactiv de legare simplă a datelor, Vue are suport și pentru legare reactivă bidirecțională a datelor. Legarea bidirectionala se întâmplă atunci când datele sunt, de asemenea, legate și de DOM înapoi la Javascript. De exemplu, schimbarea valorii unui mesaj din DOM printr-un element de tipul input. Doar trebuie adăugat documentului un input cu un atribut v-model care are atribuită proprietatea mesajului respectiv.

De exemplu:



Dacă se va introduce ceva în inputul folosit, variabila din Javascript va fi actualizată în același timp cu modificarea, care la rândul său actualizează elementul de tip h1 folosit.

Atributul v-model este o directiva Vue. Scopul unei directive este de a aplica reactiv un comportament special DOM-ului atunci cand valoarea sa se modifică. În căzut directivei v-model, leagă elementul de una dintre proprietățile datelor pe care Vue le urmărește în permanență.

### 1.1.3 Sintaxa sistemului de legare a datelor

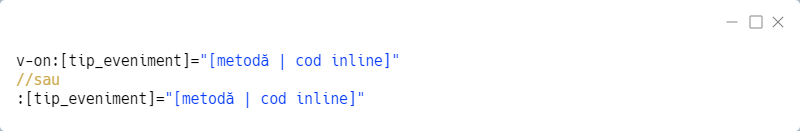
Vue.js acceptă interpolare de text folosind acolade duble, de exemplu: {{mesaj}}. Acest tip de sintaxa se numește sintaxa mustață.

În timpul compilării, tag-ul {{mesaj}} va fi înlocuit cu valoarea mesajului din obiectul de date și va fi actualizat în timpul rulării ori de cate ori se modifica valoarea titlului.

Directiva v-bind ajută la legarea atributelor sau a unei componente de o expresie a unei proprietăți de date. Legarea se poate declara cu sintaxa: v-bind[atribut] sau :[atribut], o forma prescurtata.

Presupunand ca exista o valoare numita link în care este stocat un link către o pagina web, un element de tip ancora poate fi scris astfel:



Directiva v-on este, împreună cu directiva v-bind, una dintre cele mai simple directive folosite în toate aplicațiile Vue. V-bind doar leaga un atribut de proprietatea unei date, v-on atașează un event listener elementului și leaga acel eveniment de o metodă disponibilă (sau cod scris inline):

Este bine de știut ca v-on poate doar asculta evenimentele DOM native numai atunci cand este folosit pe elemente HTML normale. In acest caz, metoda va primi evenimentul nativ ca argument și va avea acces la evenimentul $event dacă se vor folosi declarații inline.

### 1.1.4 Componentele din Vue.js

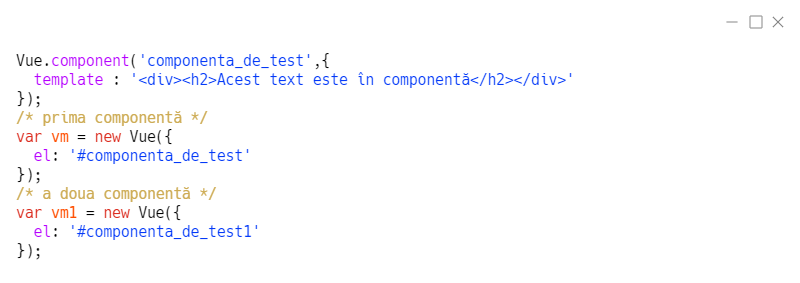
Componentele Vue sunt una dintre cele mai importante caracteristici ale framework-ului Vue.js care creeaza elemente personalizate care pot fi refolosite cu usurinta in HTML precum paragrafe, tabele, headere și footere, etc. Scopul acestor componente este de a organiza proiectul mai bine, de a face codul mai ușor de citit, și de a elimina necesitatea refolosirii codului pe diferite pagini.

Exemplu:

Partea HTML:



vue\_componenta.js



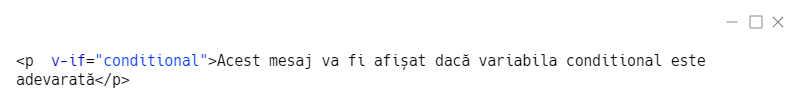
În acest exemplu, în fișierul html sunt create doua elemente div cu id-ul componenta\_de\_test și componenta\_de\_test1, iar în fișierul javascript, sunt create doua instanțe Vue cu id-urile div-urilor. În componenta creată în fișierul javascript există un șablon căruia i s-a atribuit cod HTML.

Aceasta este o metodă de a înregistra o componentă global, accesibilă ușor, fiindcă face parte din orice instanță Vue. Componenta astfel poate fi refolosită, apelată și afișată din orice fișier de tip .vue care aparține proiectului, prin sintaxa <nume\_componentă></nume\_componentă>.

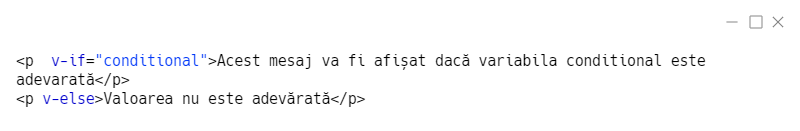
### 1.1.5 Directive Vue

Pe lângă directivele prezentate până acum, mai sunt folosite des directivele v-if și v-for.

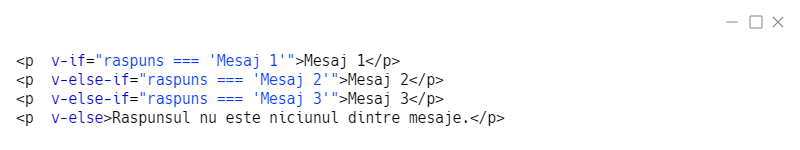
Directiva v-if este folosită pentru a afișa o parte de cod în mod condițional (codul va fi afișat doar dacă expresia directivei returnează o valoare adevărată):



De asemenea, se poate adăuga o directiva v-else, în caz ca variabila nu este adevărată:

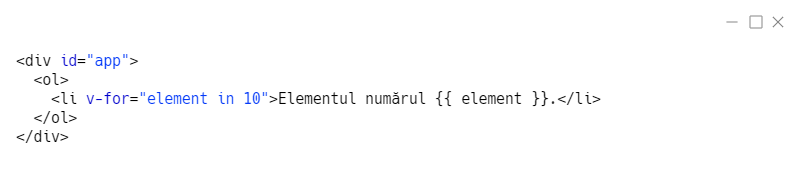


Altă posibilitate este adăugarea directivei v-else-if, în caz ca partea de cod este mai complexă:



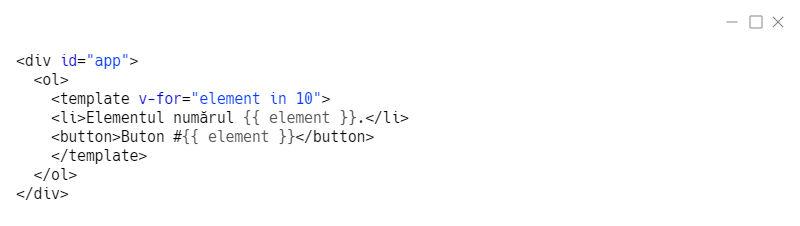
Directiva v-for este folosită pentru a afișa o listă de elemente bazate pe un vector sau o matrice. V-for necesita o sintaxa diferită de celelalte directive de până acum, de tipul for element in elemente, unde elemente reprezinta sursa listei de date, iar element este un nume pentru fiecare element peste care se itereaza atunci cand se afișează lista.

Presupunând că se dorește să se afișeze o lista ordonata cu numere de la 1 la 10. O soluție pentru aceasta problema este:

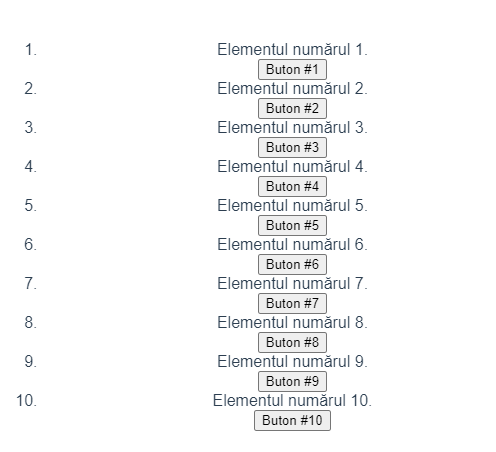


Aceste linii de cod vor afișa lista dorită sub forma: Elementul numărul 1, Elementul numărul 3, Elementul numărul 4, etc.

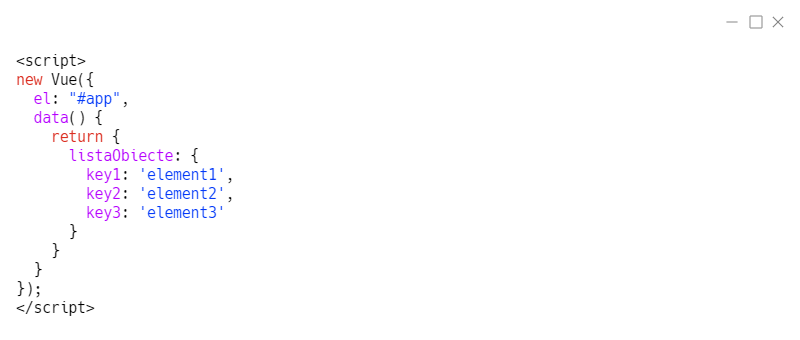
Directiva v-for se aplica doar elementului careia ii este atașată. Dacă se dorește ca mai multe elemente să se repete cu directiva v-for, acestea trebuie puse între tag-uri de tip <template>. De exemplu, dacă se dorește să se afișeze câte un buton pentru fiecare element din lista creată anterior, acest lucru se poate face astfel:



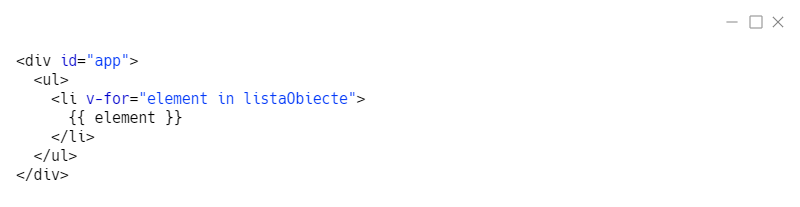
Rezultat:



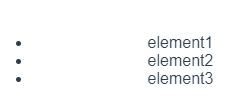
Lucrul cu obiecte și cu v-for nu este foarte diferit. Putem itera peste valorile dintr-o listă din modelul de date. De exemplu, așa arată obiectul de date cu aceste modificări:



Și așa arată acum partea de HTML:

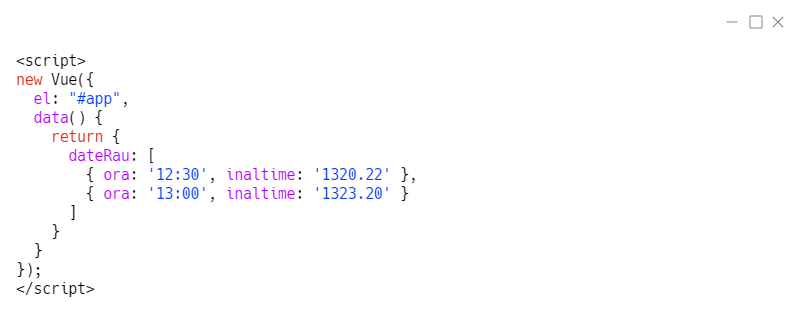


Cu rezultatul:

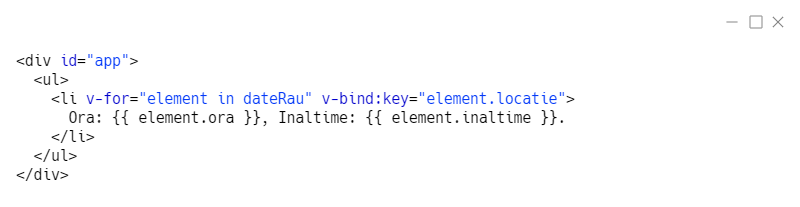


V-for poate fi folosit pentru a afișa elemente dintr-un obiect, într-un mod similar cu cel de până acum. Pentru o iterare mai complexă, se poate folosi atributul v-bind:key sau :key.

Codul din data object:



Codul din partea HTML:



Ca rezultat, pe pagina web se vor afișa toate elementele din obiect.

### 1.1.6 Rutarea componentelor

Vue.js oferă o mulțime de funcții care permit crearea componentelor web reutilizabile, iar rutarea este una dintre aceste metode. Aceasta permite utilizatorului să comute între diferite pagini fără a actualiza pagina respectivă.

Presupunând că se dorește crearea unei rute către /home și una către pagina /settings, primul pas este modificarea fișierului index.js, aflat în folderul router al proiectului. Pentru a importa pagina Settings.vue se poate folosi una dintre următoarele sintaxe:



Astfel, așa va arăta fișierul după modificări:



Apoi, în folderul views trebuie creat fișierul Settings.vue care va contine pagina web în sine, iar în fișierul App.Vue trebuie adăugată o rută către această pagină, de exemplu:



De asemenea, în Vue pot fi integrate și rutere externe, precum Router, Director sau Page.js.

## 1.2 Vuetify

Vuetify este o biblioteca de tip Vue UI care contine componente ce urmăresc standardul designului material. Scopul acestui framework este de a oferi dezvoltatorilor instrumentele de care au nevoie pentru a construi website-uri într-un mod simplu. Spre deosebire de alte framework-uri, Vuetify este conceput pentru a fi ușor de folosit și conține sute de componente materiale.

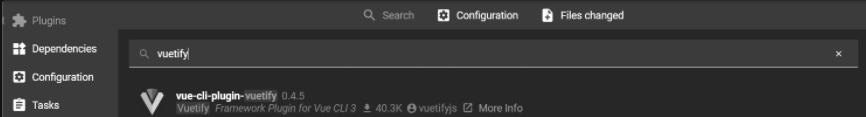
Aceasta biblioteca adopta o abordare mobile-first, lucru care ușurează munca dezvoltatorilor fiindcă aplicațiile funcționează imediat pe orice tip de platforma (computer desktop, tabletă, telefon).

### 1.2.1 Instalare

Vuetify poate fi instalat folosind Vue CLI prin navigarea către proiect și scrierea următoarelor comenzi în terminal:



Sau, din Vue UI, cautand vuetify in lista de plugin-uri:



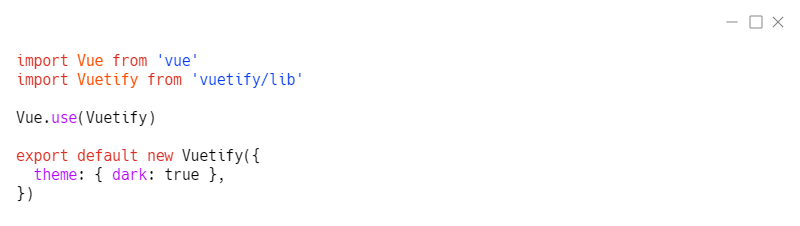
### 1.2.2 Tipuri de teme și configurarea acestora în Vuetify

Temele întunecate (dark) au devenit din ce în ce mai populare în ultimii ani, acestea afișează suprafețe întunecate pe majoritatea interfețelor și au fost concepute ca și alternativa pentru temele implicite (sau light). Acestea reduc luminanta emisă de ecranele dispozitivelor, îndeplinind în același timp raporturi minime ale contrastului. Temele întunecate ajuta la imbunatatirea ergonomiei vizuale prin reducerea oboselii ochilor (din moment ce nu conține culori deschise și stridente, sunt mai plăcut și mai ușor de privit), și facilitează utilizarea ecranului în locuri întunecate, toate acestea conservand notabil energia bateriei.

Dispozitivele care au ecrane OLED beneficiază de posibilitatea de a dezactiva pixelii de culoare neagră independent, conservand astfel și mai multă energie.

Vuetify acceptă atât variantele deschise (light), cât și cele întunecate (dark) ale specificațiilor din material design. Aceste teme sunt acceptate de majoritatea componentelor, iar în mod implicit aplicația va utiliza tema light.

Pentru a suprascrie setările implicite, trebuie modificat fișierul vuetify.js, aflat în folderul plugins al proiectului:



Cand o componenta este desemnată ca fiind deschisa sau întunecată, toți copiii ei vor moșteni și vor avea același tip de tema, dacă nu se specifică altfel.

Implicit, Vuetify are o tema standard aplicată pentru toate componentele. Acest lucru poate fi schimbat ușor: doar trebuie adaugata o proprietate tematică constructorului Vuetify. Se pot modifica independent culorile, sau pot fi schimbate pentru a folosi culori predefinite de tip material. În spate, Vuetify genereaza clase CSS bazate pe aceste valori care vor fi accesibile în DOM.

De asemenea, functionalitatea de tematică poate fi dezactivată în următorul mod:



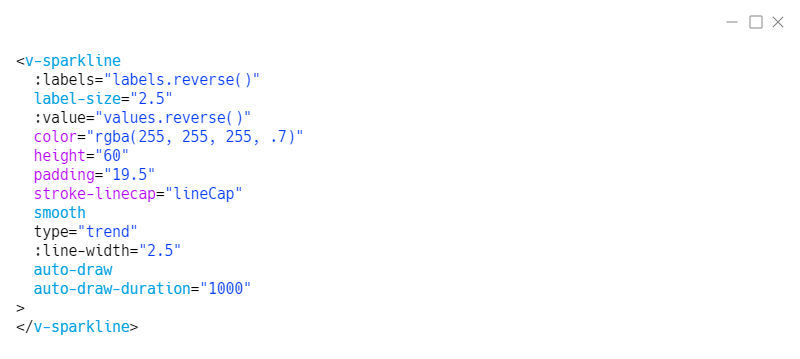
### 1.2.3 Componente Vuetify

Componenta sparkline poate fi utilizata pentru a crea grafice simple care ofera o reprezentare vizuala a datelor. Exista doua variante pentru aceasta, trend (valoare implicită), și bar. Fiecare accepta o multitudine de opțiuni pentru personalizarea aspectului.

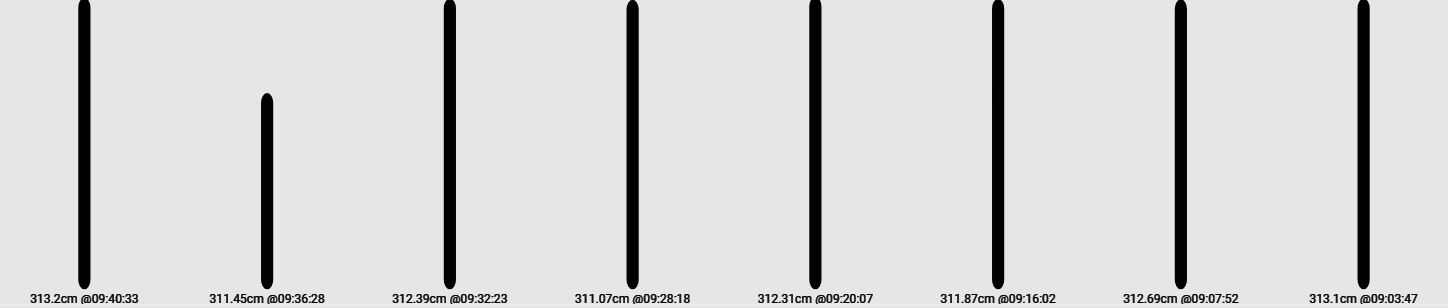
Exemplu de utilizare a componentei sparkline:



Acest grafic este creat cu următoarele proprietăți:

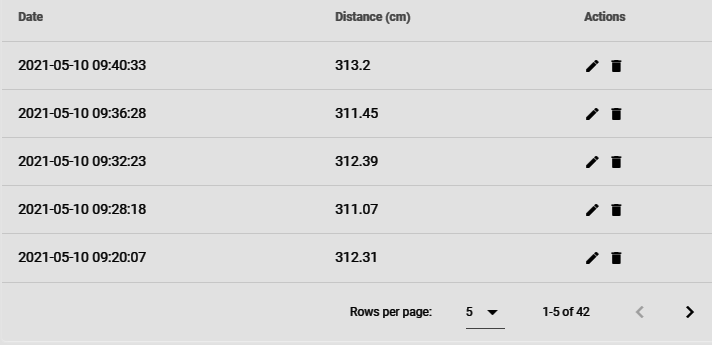


* v-sparkline este declararea componentei
* :labels reprezintă textul afișat în partea de jos (distanța și timpul de exemplu), sub forma de lista în data object
* label-size modifica mărimea fontului folosit pentru etichete
* value reprezintă o listă cu numere din data object, folosite pentru crearea graficului
* color aplica culoarea specificată, poate fi numele culorii de tip material(de exemplu yellow) sau poate fi specificata sub forma de cod hex sau rgba (ca în exemplu)
* smooth reprezintă numărul de pixeli utilizat pentru colturi; valoarea true este de 8 pixeli, iar valoarea false este de 0 pixeli
* auto-draw aplica o animație asupra graficului atunci cand acesta este încărcat, de la stanga la dreapta
* auto-draw-duration modifica numărul de secunde în care este activă animatia de auto-draw
* type specifică tipul graficului (trendline sau bar); de exemplu, așa arată graficul de mai sus cu type bar:



Componenta data table (v-data-table) este folosită pentru a afișa datelor ca tabel. Principalul motiv pentru care aceasta este superioară componentei simple table este fiindcă v-data-table include funcționalități precum sortarea, paginarea, editarea continutului tabelelor, căutarea și selectarea rândurilor. V-data-table acceptă acțiuni CRUD (create, read, update, delete) pentru fiecare rand din tabel.

Exemplu de utilizare a componentei v-data-table:



Acest tabel a fost creat cu următoarele proprietăți:

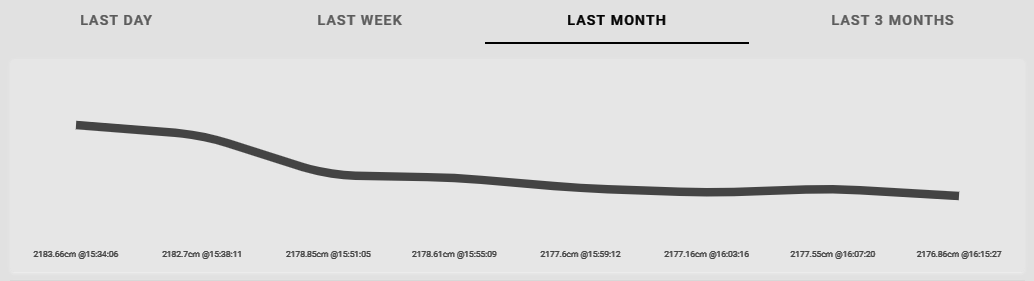


* :headers este o lista de obiecte, unde fiecare obiect descrie antetul unei coloane
* :items este lista de elemente care trebuie afișate
* update:sort-desc schimba direcția în care este făcută sortarea
* :items-per-page specifica numărul de rânduri care pot fi vizibile în pagină

Componenta v-tab este utilizata pentru ascunderea conținutului în spatele unui element selectabil, aceasta poate fi folosită ca și navigare pentru o pagina folosind routere, unde filele sunt linkuri, iar elementele tab-urilor sunt conținutul.

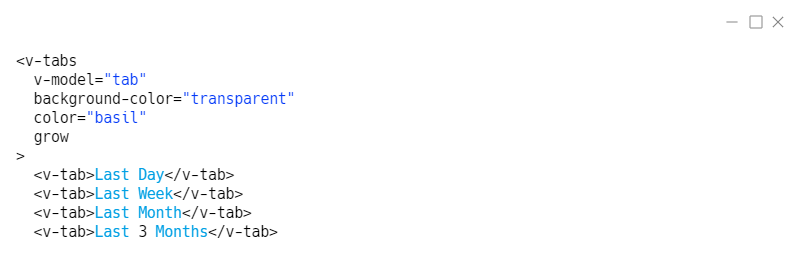
Exemplu de utilizare a componentei v-tab:





În aceste exemple, fiecare din tab-uri schimba componenta sparkline creată anterior în funcție de perioada de timp care este selectată.

Tab-urile au fost create cu următoarele proprietăți:

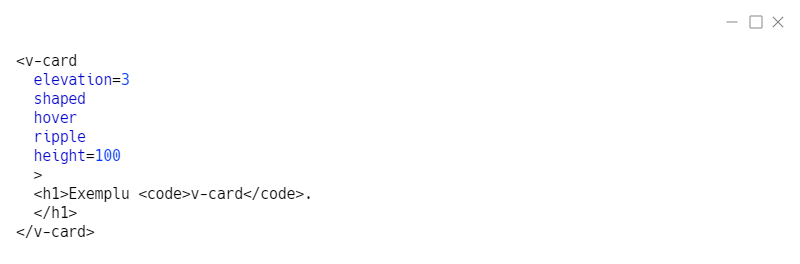


* background-color specifica culoare fundalului
* grow face ca elementele din filă să ocupe tot spațiul disponibil în lățime

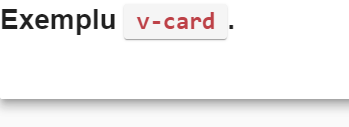
Apoi, conținutul filelor este specificat cu următoarea sintaxă (last Day lastWeek sunt componente Vue deja declarate, conținând sparklines și datele necesare pentru acesta):

 Componenta v-card din frameworkul Vuetify este o utilitate versatila care poate fi utilizata într-o mulțime de moduri (de la un panou la o imagine). Aceasta este o modalitate de a grupa conținutul similar, indiferent de platformă utilizată (web, computer, mobil). Ca și în cazul tuturor componentelor Vuetify, v-card este foarte personalizabilă. Principalul mod de a personaliza componentele este prin props (un atribut CSS personalizat).

De exemplu, dat fiind următorul cod:



Se va afișa următorul v-card:

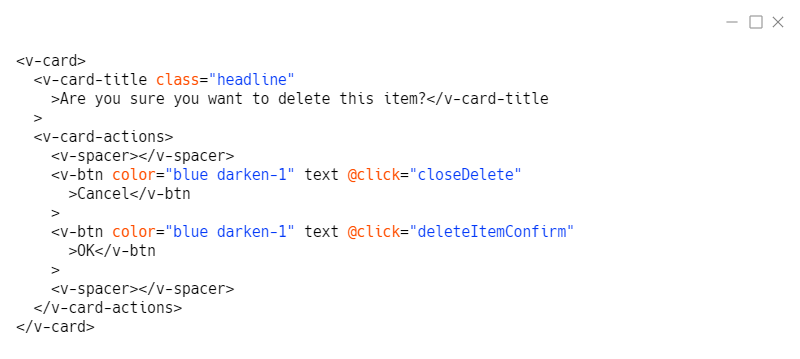


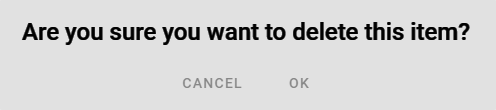
* elevation desemnează o altitudine aplicată componentei între 0 și 24 (schimbările de altitudine/înălțime a componentelor permit controlarea adancimii relative sau distanța dintre doua suprafețe de-a lungul axei verticale. Exista un total de 25 de niveluri de altitudine). Dacă se dorește eliminarea altitudinii, se poate aplica prop-ul outlined (acesta adaugă și o margine subțire cardului)
* shaped aplică o margine în partea de stanga sus și in dreapta jos a cardului
* hover aplică o înălțime de 4dp atunci cand se afla cursorul peste element (înălțimea are valoarea implicită de 2dp).
* ripple adaugă directiva v-ripple (utilizata pentru a arata acțiunea unui utilizator sub forma de animație. Poate fi aplicată oricărui element la nivel de bloc, iar un număr mare de componente vin cu aceasta directiva deja incorporata, de exemplu v-tabs-item, v-btn, etc)
* height definește înălțimea componentei în pagină

Un card are 4 componente de baza: v-card-title, v-card-subtitle, v-card-text și v-card-actions.

* v-card-title oferă o dimensiune a fontului folosit și padding pentru titlurile cardurilor
* v-card-subtitle oferă o dimensiune a fontului folosit și padding pentru subtitlurile cardurilor
* v-card-actions este containerul utilizat pentru gruparea acțiunilor asupra unui card (cum ar fi v-menu sau v-btn).

De exemplu, codul următor va genera un card care apare la apăsarea unui buton care șterge un rand dintr-un tabel:



Rezultat:

Componenta v-btn înlocuiește butonul standard folosit in HTML cu o tema de material design care contine o multitudine de opțiuni pentru personalizare. În forma lor cea mai simpla, butoanele contin text cu majuscule, o altitudine ușoară față de elementul din stratul de mai jos, și o animație de tip ripple la acțiunea click-ului.

Este bine de menționat faptul ca v-btn este singura componenta care se comporta diferit atunci cand i se aplica prop-ul dark. În mod normal, componentele folosesc acest prop pentru a indica faptul că au un fundal de culoare închisă și că trebuie ca textul lor să fie de culoare alba. Deși acest lucru funcționează pentru v-btn, este recomandat să se folosească prop-ul doar atunci când butonul este pe un fundal colorat, din cauza stării de dezactivare care poate fi greu de văzut pe fundaluri de culoare albă.

De exemplu, pentru acest cod:



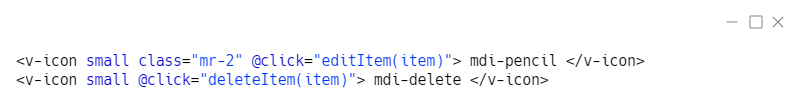
Se va afișa în pagină următorul buton:



* block extinde butonul la 100% din spațiul disponibil
* large mărește componenta
* outlined face fundalul transparent și aplică o margine subțire butonului
* plain elimină schimbarea fundalului care se întâmpla atunci când se trecea cu mouse-ul peste buton
* rounded aplică o margine mare butonului

Componenta v-icon oferă un set mare de imagini pentru a oferi context diverselor aspecte unei aplicații. Pentru a utiliza o pictograma trebuie doar scris prefixul mdi- și apoi numele pictogramei, așa cum este specificat în lista de pictograme conforma designului material.

De exemplu, acest cod va genera următoarele două pictograme:





În caz că există un eveniment de tipul click aplicat unei componente v-icon, acesta va schimba automat cursorul într-un pointer.

## 1.3 JavaScript

### 1.3.1 Descriere

JavaScript este un limbaj de scriptare pentru web. Este un limbaj interpretat, ceea ce înseamnă ca nu are nevoie de un compilator pentru a ii traduce codul precum alte limbaje (C, C++), codul acestuia ruland direct într-un browser web.

Lucrând împreună cu HTML și CSS, limbajul JavaScript este folosit pentru a crea aplicații sau pagini web, fiind acceptat de majoritatea browserelor web moderne (Google Chrome, Microsoft Edge, Mozilla Firefox, Safari, Opera, etc) pentru desktop, cât și cele de pe platforme mobile (Android, iOS).

Elementele dinamice ale paginilor web pot fi controlate folosit JavaScript. Limbajul funcționează în browsere web, și, mai recent, și pe servere web. Interfețe de programare a aplicațiilor (API) sunt, de asemenea, acceptate de JavaScript.

Modul în care funcționează un browser web este următorul:

* browserul încarcă o pagină, analizează codul HTML și creează DOM-ul (Document Object Model) din conținutul acestuia, apoi DOM-ul afișează o vizualizare în timp real a paginii web codului JavaScript
* sunt preluate toate elementele legate de HTML, precum imagini și fișiere CSS (informațiile despre fișierele CSS provin de la analizatorul CSS)
* HTML și CSS sunt puse împreună de DOM pentru a crea pagina web
* motorul JavaScript încarcă fișierele .js și codul inline, dar nu le rulează imediat, ci așteaptă întâi terminarea încărcării elementelor HTML is CSS
* codul JavaScript este executat în ordinea în care este scris, acest lucru duce la actualizarea DOM-ului cu acest cod și redarea lui în browser

### 1.3.2 Vue și JavaScript

După cum ii spune numele, Vue.js este un framework pentru JavaScript, ceea ce înseamnă ca se poate scrie cod JavaScript în proiectele Vue. În fiecare fișier .vue există o parte de cod, scrisă între tag-uri de tip <script></script> cu sintaxa următoare:



* export default este folosit pentru a o înregistrare locală a unei componente Vue
* components specifica ce componente au fost importate și urmează să fie folosite în fișierul .vue
* dată este un obiect care stochează toate variabilele de care dezvoltatorul are nevoie pentru a crea pagina web; variabilele acestea sunt declarate global și pot fi numere, cuvinte, liste, alte obiecte, de tip boolean, etc. Pentru invocarea acestor variabile in HTML, se foloseste sintaxa {{ moustache }}, iar pentru folosirea lor în metodele din export default, se apelează cu prefixul this. De exemplu, o variabila numePrenume declarată în data() se apelează în metodă cu sintaxa this.numePrenume
* proprietatea computed este folosită în momentul în care codul inline devine complex; diferența dintre o funcție definită în proprietatea computed sau definită ca și o funcție în proprietatea methods este ca proprietățile computed sunt stocate in cache pe baza dependențelor lor reactive, o proprietate calculata se va reevalua numai atunci cand una sau mai multe dintre dependențele sale s-au schimbat. Un exemplu al unei proprietăți computed care nu se va actualiza niciodată este următoarea, fiindcă Date.now() nu este o dependență reactivă:



* watch: Vue oferă un mod mai generic de a observa și de a reacționa la schimbările de date dintr-o instanța Vue: proprietățile de tip watch. Deși proprietățile computed sunt mai potrivite în majoritatea cazurilor, exista momente în care este necesar un watcher personalizat. Acestea sunt cele mai utile cand se efectuează operațiuni asincrone (De exemplu, presupunem ca o aplicație web execută o parte de cod mai complexă. Motivul pentru care clientul poate continua utilizarea paginii este faptul ca funcția care contine acest cod este declarată ca fiind asincronă). Este important de menționat faptul ca metodele scrise în watch rulează imediat în momentul în care componenta în care se află este inițializată, este bine să fie cunoscut acest lucru pentru a evita problemele de DOM, cum ar fi accesarea acestuia înainte să fie montat
* methods contine toate funcțiile/metodele scrise de dezvoltator

### 1.3.3 Calcularea datei

Obiectele JavaScript de tip Date reprezinta un moment unic în timp într-un format independent de platforma, acestea conțin un numar care reprezinta cate milisecunde au trecut de la data 1 ianuarie 1970 UTC.

Abrevierea UTC provine de la Coordinated Universal Time (Timp coordonat universal) și este un standard folosit pentru a seta toate fusurile orare din întreaga lume. De exemplu, orașul București se află în fusul orar UTC+3, ceea ce înseamnă ca este cu 3 ore mai târziu în București decat citirea de pe un ceas UTC, cu excepția orei de vară din România, când este doar cu 2 ore mai târziu.

Conceptul din spatele UTC a fost conceput la sfarsitul anilor 1800, când liniile feroviare și de transport maritim conectează lumea și erau necesare orare standard pentru coordonarea activităților economice care se foloseau de aceste metode de transport pentru funcționare. Înainte, ora din zi era stabilită de ceasurile din locația respectivă, iar faptul ca orele locale pot diferi unele de altele cu câteva minute sau secunde împiedica, de exemplu, oamenii care întârziau și nu știau dacă vor prinde trenul. Acest concept a fost adoptat ca și standard în anul 1967 de un comitet de la Națiunile Unite, încorporând și măsurători ale rotației Pământului.

Data maximă din JavaScript nu are aceeași valoare cu numărul întreg maxim (Number.MAX\_SAFE\_INTEGER este 9,007,199,254,740,991). În schimb, este de ±100.000.000 (o sută de milioane) de zile fata de 1 ianuarie 1970 UTC (20 aprilie 271821 î.Hr. sau 13 septembrie 275760 d.Hr.), ceea ce este echivalent cu ±8,640,000,000,000,000 de milisecunde.

Există mai multe metode în JavaScript pentru a obține o data în diferite formate, precum și pentru a efectua conversii de fus orar. Cele mai des folosite sunt funcțiile care genereaza data și ora în Timpul Universal Coordonat (cunoscută istoric sub numele de Greenwich Mean Time, deoarece UTC se află de-a lungul meridianului care include Londra și Greenwich).

Constructorul Date(), cand este apelat ca și funcție, returnează o reprezentare sub forma de string a datei și orei curente, identic cu rezultatul codului new Date().toString().

# Capitolul II - Tehnologii pentru colectare de date

## 2.1 Raspberry PI

Raspberry PI este numele unei serii de computere construite de Raspberry PI Foundation, o organizație caritabilă din Marea Britanie formată în anul 2009 care își propune să educe oamenii în informatica și să creeze un acces la ușor la educația informatică.

Primul Raspberry PI a fost lansat în anul 2012 și de atunci au fost lansate mai multe iterații și variații ale acestuia. Modelul original PI avea un procesor single-core de 700 MHz și 256MB RAM, iar cel mai recent model (Raspberry Pi 4 model B) are un procesor Quad core Cortex A72 pe arhitectura ARM v8 de 1.5GHz, variante de 2GB, 4GB sau 8GB de LPDDR4-3200 SDRAM, 2.4GHz și 5.0GHz IEEE 802.11ac conexiune wireless, Bluetooth 5.0, Gigabit Ethernet, etc.

În întreaga lume, oamenii folosesc calculatoare Raspberry PI pentru a-și dezvolta abilitățile de programare, pentru a construi proiecte personale hardware (precum emulatoare, media servere, servere în sine, etc), pentru automatizarea casei și chiar pentru unele aplicații industriale. Cheia succesului imediat pentru Raspberry PI Foundation a fost prețul extrem de scăzut, sub 35£. Chiar dacă este un calculator atat de ieftin, poate rula sisteme de operare de baza, precum Linux. Primul model avea conexiuni USB și HDMI și mai mulți pini de intrare/ieșire de uz general (GPIO) care permit utilizatorilor să adauge senzori și alte accesorii pentru a extinde funcționalitatea.

A câștigat rapid popularitate în rândul consumatorilor ca înlocuitor de PC cu costuri reduse, dar a revigorat și entuziasmul din scena în care utilizatorii puteau construi dispozitive cu un Raspberry PI ca și computer de bază. După lansare, nu a trecut mult timp pana cand întreprinderile s-au interesat de acest minicalculator, au reușit să construiască prototipuri și să facă teste folosind aceste dispozitive cu costuri reduse, economisind potențial sute de mii de euro în urma proiectării și fabricării circuitelor integrate specifice aplicației respective.

### 2.1.1 Raspberry PI 4 Model B

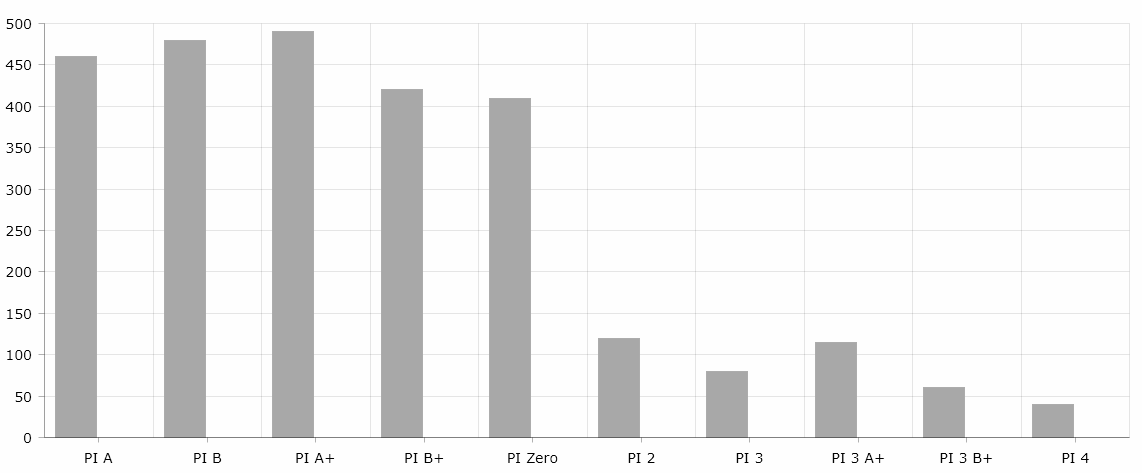
Au avut loc multe schimbări în proiectarea modelului PI 4, inclusiv prima modificare a aspectului modelului B de la lansarea Modelului B+ de Raspberry PI. Schimbarea porturilor a mărit ușor dimensiunea plăcii, măsurată în cele mai largi puncte, inclusiv porturile. Cu toate acestea, este cu 46 de grame mai ușor decat Raspberry PI 3 Model B+, probabil ajutat de lipsa portului HDMI de dimensiuni complete.

Sistemul pe un cip este acum produs pe un nod de proces de 28nm, numărul de nuclee este neschimbat, dar procesorul a fost mutat pe noul Arm Cortex-A72 care rulează la o frecvență puțin mai mare de 1,5 GHz. Cu toate acestea, placa grafică a văzut cea mai mare schimbare: fiecare Raspberry PI de până acum a folosit ca și GPU placa Broadcom VideoCore IV, în timp ce Raspberry PI 4 a trecut la o varianta personalizată a plăcii Broadcom VideoCore VI cu un motor de scanare împrumutat de la VideoCore V.

La suprafață, functionalitatea de rețea este neschimbată: Wi-Fi 802.11ac, o actualizare la Bluetooth 5.0, și un port Ethernet Gigabit prin cablu. Modelele mai vechi de Raspberry PI portul de Ethernet comunicau cu circuitul integrat printr-o banda comuna de USB 2.0, dar modelul Pi 4 se bucura de o conexiune mai directă, fără blocaje. Acest model are 4 porturi USB de dimensiuni complete: doua dintre acestea au fost actualizate la tehnologia USB 3.0, îmbunătățind foarte mult lățimea de banda teoretică disponibilă pentru dispozitivele externe de la acceleratoare conectate prin USB (precum Google Coral Edge) la simpla stocare USB. Schimbarea la USB Type-C de 3A de la micro USB-ul de 2.5A al modelelor mai vechi permite, de asemenea, conectarea dispozitivelor de putere mai mare la Raspberry fără a fi nevoie de un hub alimentat.

Un procesor mai puternic înseamnă, de obicei, mai multă căldură emanată în exces, lucru cu care cei de la Raspberry PI Foundation s-au mai intalnit in trecut: modelul B Raspberry PI 3, în special, se încălzea foarte mult. Înlocuitorul său, Raspberry PI 3 Model B+ a îmbunătățit enorm placa, folosind un PCB (printed circuit board) mai gros, un sistem pe un cip acoperit cu metal și ajustat și o tratare termica îmbunătățită.

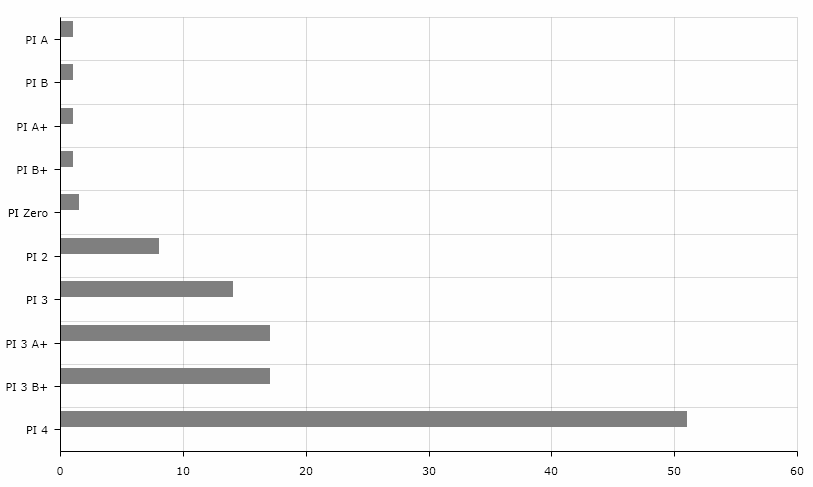
După un test de timpi realizat pe mai multe modele de Raspberry PI în care se folosea interfața de scriptare pe linia de comanda a aplicației GIMP de editare a imaginilor pentru a edita o imagine de înaltă rezoluție. Este o sarcina de lucru care se bazează atât pe performanța procesorului, cât și pe cea a memoriei RAM, cerând totodată o cantitate mare de RAM liber, lucru care penalizează modelele cu mai puțin de 1GB de RAM (în acest test se urmărește ca numărul de secunde să fie cât mai mic, reprezentând o putere de procesare mai mare):



Este clar ca diferenta de memorie RAM dintre Raspberry PI 3 Model B+ și modelul A+ mai mic și mai ieftin are un impact vizibil asupra performanței: datorită rezoluției ridicate a imaginii testate, Modelul A+ este forțat să schimbe conținutul memoriei pentru a-i face loc, lucru pe care Raspberry Pi 3 Model B+ de 1GB îl evita, și care nu pune nici o problemă modelului folosit pentru acest proiect, Raspberry PI 4 Model B cu 2GB de memorie RAM.

Deși abordarea de intrare/ieșire folosind Python asupra conexiunilor GPIO (general purpose input/output) este o sarcina obișnuită pentru Raspberry PI, următoarea comparație se intampla în cazul în care procesorul este foarte limitat: un program activează și dezactivează un pin fără pauze, iar rata la care comută pinul este măsurată folosind un contor de frecvență.

Graficul este măsurat în kHz, iar numerele mai mari sunt cele urmărite:



Nu este surprinzător faptul că performanța pinilor GPIO sunt legate de performanța procesorului. Totuși, este surprinzătoare diferența dramatică de la familia Raspberry PI 3 la Raspberry PI 4, totul datorită trecerii la noile nuclee de CPU ARM Cortex A72.

Nu se poate nega faptul că Raspberry PI 4 este un calculator impresionant. În timp ce pierderea portului HDMI de dimensiuni complete este o dezamăgire, dar faptul ca acum poate afișa simultan doua ecrane la rezoluția 4k compenseaza acest lucru pe deplin. Trecerea de la un conector USB de tip C pentru alimentare are sens, deoarece acolo funcționează piața smartphone-urilor și tabletelor de ceva timp, iar aspectul modificat va provoca probleme doar pe termen scurt, sub forma unor designuri de carcase învechite, dar pe termen lung este un câștig.

Acestea fiind spuse, nu totul este o îmbunătățire. Noul sistem pe un cip BCM 2711 degajează multă căldură, iar răcirea activă este mai importantă ca niciodată pentru proiectele care pun un Raspberry PI încărcat cu sarcini într-o zonă închisă. Redarea video in rezolutie 4K promisă este limitată la conținutul de tip H.265, în timp ce decodarea hardware pentru MPEG2, MPEG4 și H.263 a fost abandonată, în condițiile în care procesorul este suficient de puternic pentru a decoda aceste formate cu software fără a-i pune multe probleme.

Dezavantajele nu depășesc în niciun caz avantajele, iar cu Raspberry PI 4, Fundația a abordat o listă de plângeri și reclamații pe care comunitatea le-a exprimat de la originalul Raspberry PI Model B: blocajul USB a dispărut, exista porturi de mare viteza pentru dispozitive externe, exista variante cu mai mult de 1GB de memorie RAM pentru cei care au nevoie de aceasta și două ieșiri de afișaj: rezolutie 4K la 60 de cadre pe secundă pe un singur ecran, sau rezolutie 4K la 30 de cadre pe secundă, pe două ecrane. În mod neoficial, placa poate ajunge la 4K și 60 de cadre pe secundă pe două ecrane prin overclockarea plăcii video. Împreună cu lansarea unei noi distribuții de Raspberry Pi OS (Linux), bazată pe Debian Buster, Raspberry PI 4 este primul care poate, în mod realist, să înlocuiască un calculator desktop pentru o gamă largă de oameni care utilizează computere.

Există și îmbunătățiri suplimentare în acest model, principala este adăugarea de bus-uri suplimentare: există patru bus-uri seriale de tip UART, patru de tip SPI, și patru de tip I2C, care vor fi binevenite de cei care construiesc aplicații mai complicate. Funcționalitatea de modulare a lățimii impulsurilor (PWM, pulse width modulation) a fost actualizată și nu mai este partajată cu ieșirea analog audio disponibilă pe mufa AV de 3,5mm.

### 2.1.2 Sistemul de operare Raspberry Pi OS

Raspberry Pi OS (numit Raspbian în trecut) este un sistem de operare gratuit, optimizat pentru hardware-ul Raspberry Pi și este bazat pe Debian (Linux). Un sistem de operare este un set de programe de bază și utilitare care fac că Raspberry Pi-ul să funcționeze din punct de vedere software. Cu toate acestea, Raspberry Pi OS oferă mai mult decat un sistem de operare pur: are preinstalate peste 35,000 de pachete, software precompilat, pregătit pentru o instalare ușoară pe orice placă Raspberry Pi.

Prima versiune de Raspberry Pi OS de peste 35,000 de pachete, optimizată pentru cea mai bună performanță pe Raspberry Pi a fost finalizată în iunie 2012, dar sistemul de operare este în curs de dezvoltare activa, cu accent pe îmbunătățirea stabilității și performanței a cat mai multor pachete Debian.

### 2.1.3 Instalarea sistemului Raspberry Pi OS

Există mai multe metode pentru a instala Raspberry Pi OS, precum instalarea manuala a unei imagini care contine sistemul de operare, rularea acestuia de pe o memorie de tip flash, sau folosirea programului Raspberry Pi Imager, un software care oferă cea mai rapidă și simplă modalitate de a instala Raspberry Pi OS și alte sisteme de operare pe un card microSD.

Pentru început, trebuie ales un sistem de operare. Raspberry Pi Imager vine cu cateva sisteme de operare la alegere, deci nu trebuie descarcat nimic separat. După ce se alege sistemul de operare (în acest caz Raspberry Pi OS), trebuie selectat cardul SD pe care va fi instalat acesta (se recomandă să fie un card de minim 8GB), apoi se face click pe butonul write.

Trebuie menționat faptul că Raspberry Pi Imager conține două utilități folositoare, pe lângă cea de instalare a unui sistem de operare:

* EEPROM Recovery - modelele de Raspberry Pi 4 au atașate pe interfața periferică în serie (SPI - Serial Peripheral Interface) memorie EEPROM la 4Mb/512Kb (memorie read-only programabilă care poate fi ștearsă electric), care conține codul folosit pentru a porni sistemul, și înlocuiește fișierul bootcode.bin găsit anterior în partitia de boot a cardului SD pe care se face instalarea sistemului de operare (acest software care funcționează doar pe modelele de Raspberry Pi 4)
* Erase tool - o utilitate care se poate folosi pentru a formata carduri SD în formatul Fat32 (necesar pentru citirea cardului SD de către Raspberry Pi și pentru a putea instala un sistem de operare pe acesta)

De asemenea, Raspberry Pi OS poate fi rulat și de pe o memorie de tip flash, dar performanța cardului SD este foarte lentă.

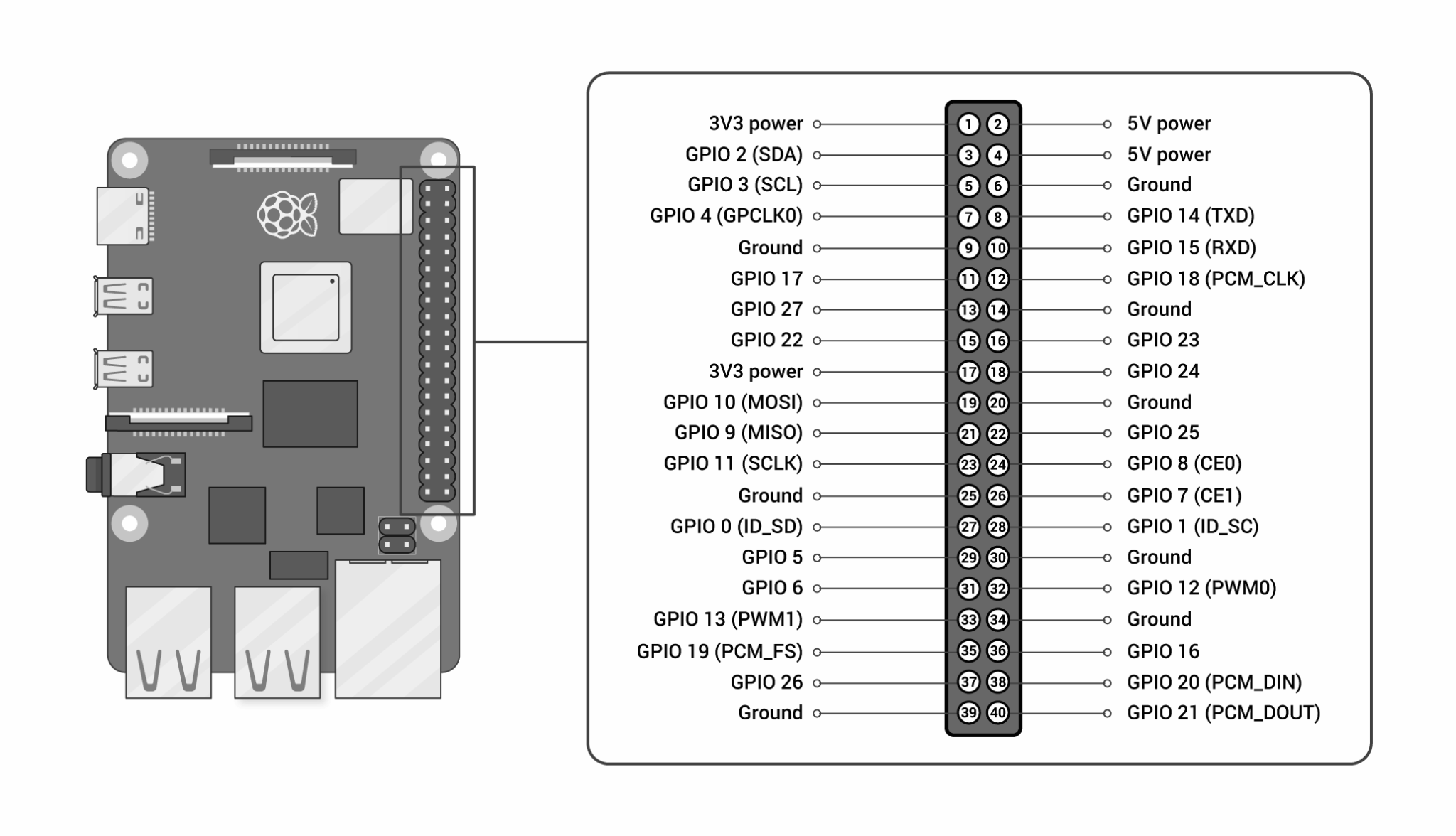
### 2.1.4 GPIO

GPIO (General Purpose Input/Output) este o interfața standard utilizată pentru conectarea microcontrolerelor la alte dispozitive electronice. De exemplu, poate fi utilizat cu senzori, display-uri, diode și module de tip sistem pe un cip.

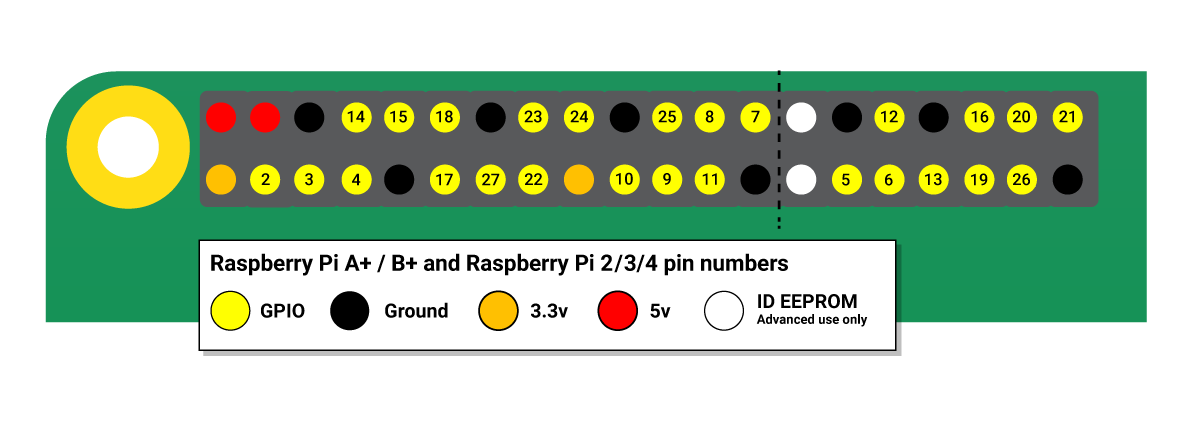
O caracteristică importantă a calculatoarelor Raspberry Pi este rândul de pini GPIO de-a lungul marginii superioare a plăcii. Pe toate plăcile Raspberry Pi actuale se găsesc 40 de pini GPIO, iar înainte de modelul Pi 1 Model B+ din 2014, plăcile aveau doar 26 de pini.

Oricare dintre pinii GPIO poate fi desemnat, folosind software, ca pin de intrare sau de ieșire și utilizat pentru o gama largă de scopuri.

Diagramă pentru pinii GPIO:



Numerotarea pinilor GPIO nu este neapărat în ordine numerică, pinii GPIO 0 și 1 sunt prezenți pe placa, ca și pinii fizici 27 și 28, dar sunt rezervați pentru utilizari avansate.

 Sunt prezenți pe placă doi pini de 5V și doi de 3V3, precum și un număr de pini de tip ground (OV), care nu pot fi configurate. Pinii rămași sunt toți pinii 3V3 de uz general (ieșirile sunt setate la 3V3, iar intrările suportă tot 3V3.

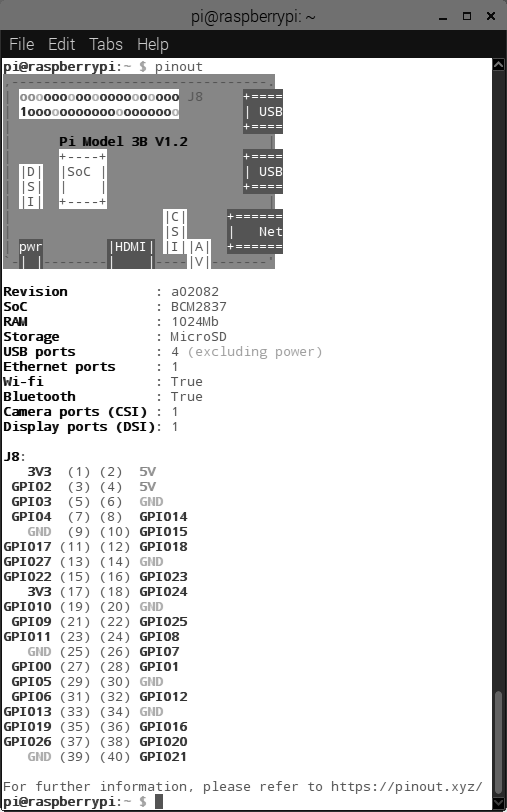
Un pin GPIO desemnat ca pin de intrare poate fi citit ca fiind high (3V3), sau low (OV). Acest lucru este făcut mai ușor prin utilizarea rezistentelor interne de tip pull-up sau pull-down. Pinii GPIO2 și GPIO3 au rezistori de tip pull-up ficși, dar pentru alți pini acest lucru poate fi configurat în software.

Pe langa utilizarea ca și dispozitive simple de intrare și ieșire, pinii GPIO pot fi utilizati cu o varietate de funcții alternative. Unele sunt disponibile pe toți pinii, alții pe pini specifici:

* PWM (pulse-width modulation): GPIO12, GPIO13, GPIO18, GPIO19
* SPI0: MOSI (GPIO10), MISO (GPIO9), SCLK (GPIO11), CE0 (GPIO8), CE1(GPIO7)
* SPI1: MOSI (GPIO20), MISO (GPIO19), SCLK (GPIO21), CE0 (GPIO18), CE1 (GPIO17), CE2 (GPIO16)
* I2C: Data (GPIO2), Clock (GPIO3), EEPROM Data (GPIO0), EEPROM Clock (GPIO1)
* Serial: TX(GPIO 14), RX(GPIO15)

În caz că utilizatorul dorește să vadă pinout-ul pinilor GPIO, pot scrie comanda pinout în terminal. Aceasta utilitate este data de librăria de Python GPIO Zero, care este preinstalată în Raspberry PI OS.

Rezultatul rulării comenzii pinout în terminal:



Pinii GPIO pot fi controlați folosind un număr mare de limbaje de programare și utilități, precum: Python, Scratch, C/C++, Processing 3, etc.

## 2.2 Python

Python este un limbaj de programare popular care poate fi utilizat pentru o mare varietate de aplicații. Include structuri high-level, legare dinamica, tastare dinamica, și multe alte caracteristici care îl fac la fel de util pentru dezvoltarea de aplicații complexe, precum și pentru scriptare sau cod care conecteaza componente. De asemenea, poate fi extins pentru a face apeluri de sistem către aproape toate sistemele de operare și pentru a rula cod scris în C sau C++. Datorită capacității sale de a rula pe aproape orice arhitectura de sistem, Python este un limbaj universal.

Acest limbaj de programare include mii de module, disponibile in Python Package Index (PyPI). PyPI oferă standarde populare pentru diferite cazuri, precum Django pentru dezvoltarea web și NumPy, Mathplotlib pentru data science.

Dezvoltat pentru prima dată la sfârșitul anilor 1980 de Guido van Rossum, Python a avansat ca limbaj de programare open source prin gestionarea discuțiilor publice prin intermediul propunerilor de impunatatire a limbajului Python (PEP). În 2018, van Rossum a renunțat la funcția de Benevolent Dictator For Life al limbajului și, așa cum este scris în PEP 13, a fost instituit un consiliu de conducere care să servească drept conducere a limbajului.

PSF (Python Software Foundation) este o corporație non-profit care deține drepturile de proprietate intelectuala din spatele limbajului de programare Python. Aceasta include versiunea Python 2.1 și ulterioară, PyPi, implementarea Python și infrastructură pentru menținerea limbajului.

În prezent, limbajul Python se află la a treia versiune majoră și este actualizat în mod regulat (cea mai recentă versiune este 3.9.5).

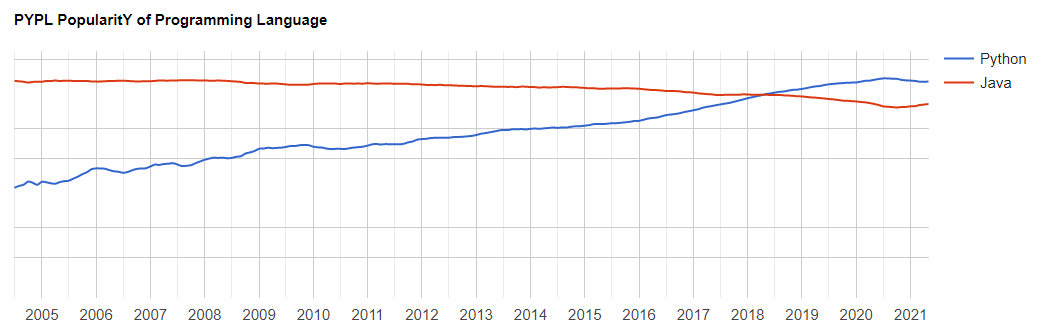
In 1999, Guido van Rossum a definit obiectivele pentru viitor pentru Python:

* un limbaj ușor și intuitiv, la fel de puternic ca și al marilor concurenți
* open source, astfel încât oricine poate contribui la dezvoltarea acestuia
* cod la fel de ușor ca limba engleză simplă
* potrivit pentru sarcinile de zi cu zi, permitand timpi scurți de dezvoltare

După 20 de ani, este clar că toate aceste obiective au fost îndeplinite cu succes. Conform PYPL (PopularitY of Programming Language Index), in acest moment Python este cel mai popular limbaj de programare:

|  |  |  |
| --- | --- | --- |
| Rang | Limbaj | Pondere |
| 1 | Python | 29.9 % |
| 2 | Java | 17.72 % |
| 3 | JavaScript | 8.31 % |
| 4 | C# | 6.9 % |
| 5 | C/C++ | 6.62 % |
| 6 | PHP | 6.15 % |
| 7 | R | 3.93 % |
| 8 | Objective-C | 2.52 % |
| 9 | Swift | 1.96 % |
| 10 | TypeScript | 1.89 % |

La nivel mondial, Python a crescut cel mai mult în ultimii 5 ani (17.0 %) și Java a scăzut cel mai mult (-6.7 %).



### 2.2.1 Avantaje ale limbajului Python

Există o mulțime de motive pentru care Python este o alegere buna ca și limbaj de programare, în funcție de ce doresc dezvoltatorii.

Dezvoltatorii noi în lumea programării pot beneficia de nivelul ridicat de abstractizare al limbajului Python. Este extrem de atractiv și este cunoscut pentru sintaxa specială (inclusiv spațiile din cod - whitespace). Python, ca și alte limbaje de nivel înalt, are un proces de garbage collection pentru a gestiona memoria sau a șterge resursele neutilizate. Un utilizator poate primi feedback instant de la interpretor, scriind python pe linia de comanda. Mulți dezvoltatori apreciază, de asemenea, faptul că Python are o sintaxă strictă impusă de compilator, ceea ce face mai ușor să existe un singur mod corect de a scrie un program.

Indiferent de nivelul lor de experiența, programatorii din multe medii diferite contribuie la limbaj în moduri semnificative. Python are un ecosistem de instrumente gratuite și proprietare, inclusiv medii integrate de dezvoltare (IDE) și framework-uri.

Poate cel mai important aspect al limbajului, Python are o comunitate enorma de utilizatori. Acest lucru înseamnă că, indiferent de problema pe care o are un dezvoltator cu Python, este posibil să existe deja oameni care lucrează sau au găsit deja o soluție la aceasta problema. Există numeroase medii de dezvoltare și alte instrumente din care se poate alege și mii de pachete open source disponibile pentru Python.

Exista multe organizații care utilizează Python. Unele dintre cele mai cunoscute companii sunt: Google, Mozilla, Microsoft, Netflix, Uber, Reddit, Dropbox, Slack, Digital Ocean, Lyft, Sauce Labs, Fastly, CapitalOne, Bloomberg, JPMorgan, etc. În plus, multe tehnologii de infrastructura IT sunt scrise in Python. Este limbajul principal utilizat pentru proiectul masiv de cloud computing OpenStack, care alimentează serviciile cloud private și publice în centrele de date din întreaga lume.

### 2.2.3 Module

Un modul permite organizarea logica a codului scris în Python. Gruparea codului aferent într-un modul facilitează înțelegerea și utilizarea codului. Un modul este un obiect Python cu atribute numite în mod arbitrar. În cuvinte simple, un modul este un fișier format din cod Python. Acesta poate defini funcții, clase și variabile (de asemenea, poate include și cod executabil).

Se poate utiliza orice fisier Python ca un modul, scriind următorul cod pentru importarea acestuia:



Presupunând că modulul are declarată și implementată în el o funcție de printare numita display, aceasta se poate apela astfel după importare:



Atunci cand interpretorul intalneste o declarație de import, acesta importă modulul dacă este prezent în calea de căutare (o cale de căutare este o lista de directoare pe care interpretorul le verifica înainte de a importa un modul).

Un modul este încărcat o singura data, indiferent de cate ori este importat. Acest lucru împiedică executarea modulului să se întâmple din nou dacă apar importări multiple.

Instrucțiunea from din Python permite importarea unor atribute specifice dintr-un modul în namespace-ul curent. De exemplu, presupunand ca dorim să importăm funcția arie din modulul modul1:



Aceasta instrucțiune nu importă întregul modul modul1 în namespace-ul curent, doar introduce funcția arie din modul.

De asemenea, se poate importa un modul întreg în namespace folosind următoarea instrucțiune:



Aceasta este o modalitate ușoară de a importa tot ce contine un modul.

Module:

#### 2.2.3.1 PiCamera

Modulul picamera este o camera portabila ușoară care poate fi folosită cu Raspberry PI. Comunică cu acesta utilizând protocolul MIPI de interfață serială a camerei și este utilizat în mod normal în procesarea imaginilor, machine learning sau în proiecte cu sisteme de supraveghere. Raspberry PI poate folosi și camere web normale conectate prin USB, dar modulele picamera sunt mai ușor de integrat și de folosit în programe.

După importarea acestui modul în codul nostru Python, avem posibilitatea de a face și manipula poze și de a înregistra videoclipuri. De exemplu, dacă se dorește să apară text pe o poza capturată, se poate folosi următoarea secvență de instrucțiuni:



Poza se va salva in folderul photos, având ca nume data și ora curentă, va avea rezolutia de 500 x 500 px, iar textul va avea culoarea alba și outline negru.



#### 2.2.3.2 Time

Modulul time în Python oferă multe modalități de reprezentare a timpului în cod, cum ar fi obiecte, numere și stringuri. De asemenea, oferă și alte funcții decât reprezentarea timpului, cum ar fi așteptarea în timpul executarii codului și măsurarea eficienței codului.

Functia sleep() suspenda execuția firului curent de execuție pentru un număr dat de secunde.

De exemplu:



* programul afișează primul print instant
* se suspendă execuția programului pentru 3.1 secunde
* se afișează și al doilea print

#### 2.2.3.3 Datetime

O dată in Python nu are un tip de date propriu (precum numerele sau caracterele), dar se poate importa un modul numit datetime pentru a lucra cu date ca obiecte.

De exemplu:



Când se vor executa instrucțiunile de mai sus, rezultatul va fi unul de acest tip:



Data conține anul, luna, ziua, ora, minutul, secunda și microsecunda curentă.

#### 2.2.3.4 RPi.GPIO

Raspberry-gpio-python sau RPi.GPIO este un modul Python pentru a controla interfața GPIO de pe Raspberry Pi.

Modulul este instalat implicit pe versiunile recente de Raspbian Linux. Pentru a utiliza modulul în programe Python, acesta trebuie mai întâi importat, folosind comanda:



Astfel se poate face referire la toate funcțiile din modul folosind numele mai scurt GPIO.

RPi.GPIO acceptă referirea la pinii GPIO utilizând fie numerele pin-urilor fizice de pe conectorul GPIO, fie numele canalelor BCM de la Broadcom SOC la care sunt conectati pinii. De exemplu, pinul 24 este pe canalul BCM GPIO8. Pentru a utiliza numerele fizice ale pinilor, trebuie folosită următoarea comandă:



Iar pentru a utiliza numerele BCM, se poate folosi: GPIO.setmode(GPIO.BCM).

Oricare dintre metode va functiona. Schema numerică BOARD are avantajul ca biblioteca cunoaște modelul de Raspberry Pi pe care rulează și va funcționa corect, chiar dacă numele canalelor Broadcom SOC se vor schimba în viitor.

Pentru a configura un canal ca intrare sau ieșire, trebuie apelat fie: GPIO.setup(canal, GPIO.IN) sau GPIO.setup(canal, GPIO.OUT), unde primul argument este numărul canalului pe baza sistemului de numerotare specificat anterior cu setmode.

Pentru a citi valoarea unui canal de intrare, trebuie apelat: GPIO.input(canal), unde argumentul este numărul canalului utilizat în configurare. Va returna valoarea 0, GPIO.LOW sau False (toate sunt echivalente) dacă valoarea este scăzută, și 1, GPIO.HIGH sau True dacă a fost citită la un nivel ridicat.

Dacă se dorește setarea unui pin GPIO pentru ieșire, se apeleaza GPIO.output(canal, stare), unde primul argument este numărul canalului și starea este nivelul de ieșire dorit: fie 0, GPIO.LOW, fie False, pentru o valoare mică, sau 1, GPIO.HIGH, sau True pentru un nivel ridicat.

În caz că există nevoia de a folosi pinii de mai multe ori într-un timp scurt, este destul de folositor să se apeleze funcția sleep() din modulul time pentru cateva secunde, pentru a evita uzura pinilor și a modulelor conectate prin intermediul pinilor General Purpose Input/Output.

Cand nu se mai folosește acest modul, este bine să se elibereze orice resurse utilizate și să se readucă toate canalele să valorile implicite de a fi intrări. Acest lucru se face apelând GPIO.cleanup().

#### 2.2.3.5 base64

Acest modul oferă funcții pentru codificarea datelor binare în caractere ASCII imprimabile și decodificarea acestor codificări înapoi la date binare.

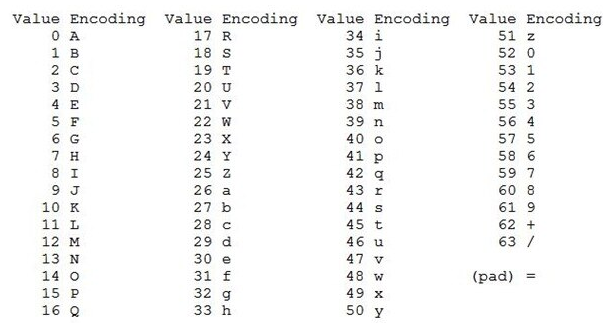
Oferă funcții de codificare și decodare pentru codificările specificate în RFC 3548, care definește algoritmii Base16, Base32 și Base64, precum și pentru codificările Ascii85 și Base85.

Există două interfețe furnizate de acest modul. Interfața modernă acceptă codarea obiectelor asemănătoare octeților la octeți ASCII și decodarea obiectelor asemănătoare octeților sau șirurile care conțin ASCII în octeți. Ambele alfabete de bază-64 definite în RFC 3548 (normale și sigure pentru URL și sistem de fișiere) sunt acceptate. Interfața legacy nu acceptă decodarea din șiruri, dar oferă funcții pentru codificare și decodare către și din obiecte de fișiere. Acceptă doar alfabetul standard Base64 și adaugă linii noi la fiecare 76 de caractere.



Această secvență de instrucțiuni va codifica o imagine folosind modulul base64, aceasta codare fiind memorata in variabila encoded.

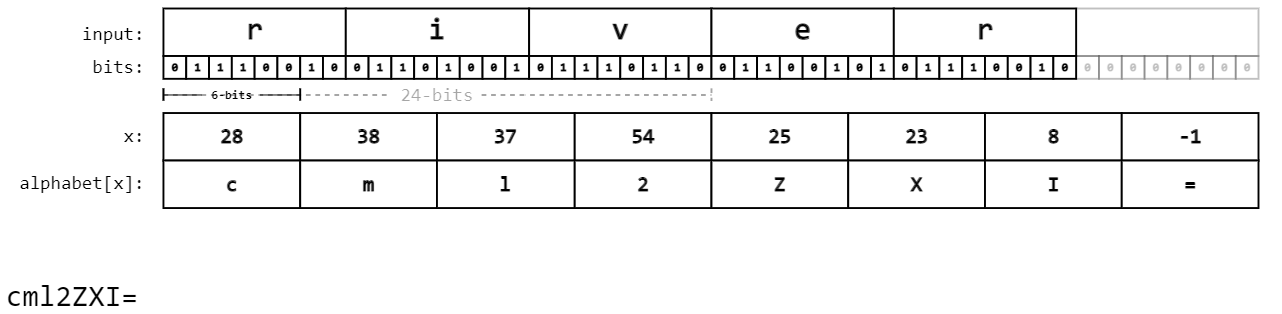
Tabelul folosit pentru codificarea în base64:



Procesul de codificare a unui flux de intrare este destul de simplu:

* fluxul de octeți este citit de la stanga la dreapta
* sunt concatenate trei grupuri de 8 biți din fluxul de intrare pentru a forma un grup de 24 de biți
* grupul de 24 de biti este tratat in continuare ca 4 grupuri de cate 6 biti, care este completat la dreapta cu zero-uri. Motivul pentru care se grupează în grupuri de cate 6 este ca 6 biti vor acoperi gama de caractere imprimabile
* fiecare dintre aceste 4 grupuri este apoi codificat folosind tabelul de codificare de mai sus

De exemplu, așa arată codificarea în base64 a cuvantului river:



## 2.3 Google Cloud Platform

Google Cloud (cunoscut și sub numele de Google Cloud Platform sau GCP) este un furnizor de resurse de calcul pentru dezvoltarea, implementarea și operarea aplicațiilor pe web. Deși infrastructura cloud servește ca gazdă pentru aplicații precum Google Workplace (fost Google Suite și anterior Google Apps), GCP este în principal un serviciu de construire și menținere a aplicațiilor originale, care pot fi apoi publicate prin web de la scara sa superioară facilitățile centrului de date.

Când rulează un site Web, o aplicație sau un serviciu pe GCP, Google ține evidența tuturor resurselor pe care le folosește: în mod specific, câtă putere de procesare, stocare a datelor, interogări de baze de date și conectivitate de rețea consumă.

Amazon și Microsoft își operează propriile platforme cloud, numite AWS și, respectiv, Azure. GCP este competitorul lor și, deși ocupă locul trei printre aceste trei în ceea ce privește cota de piață și veniturile, este un concurent solid, cu caracteristici și servicii unice, care îi oferă un avantaj în anumite scenarii.

Avantajele competitive ale GCP sunt următoarele:

* Automatizarea implementării aplicațiilor moderne. O aplicație este alcătuită din mai multe părți mobile, motiv pentru care unii dezvoltatori preferă să își construiască aplicațiile în cloud, pentru a începe. Google deține Kubernetes, care este un orchestrator pentru aplicații compuse din mai multe componente. La început, Google a adoptat o abordare proactivă pentru automatizarea implementării acestor aplicații cu mai multe fațete în cloud: de exemplu, deschizându-se către Kubo, o platformă de automatizare creată pentru a ajuta dezvoltatorii care folosesc Cloud Foundry să-și implementeze aplicațiile de pe platforme dev în cloud.
* Creative cost control. În loc să fie liderul low-cost, strategia Google cu GCP este de a permite competitivitatea costurilor în anumite scenarii. De exemplu, Google oferă un manager al ciclului de viață pentru stocarea datelor de obiecte, care permite descărcarea sau ștergerea obiectelor care nu au fost utilizate de 30 de zile sau mai mult.
* Inițiere mai prietenoasă a utilizatorilor noi. O platformă de servicii cloud poate fi un concept copleșitor pentru un nou venit. Așa cum nu a fost evident pentru mulți consumatori care a fost scopul unui microcomputer, un cloud public este o tehnologie nouă și străină pentru oamenii care sunt obișnuiți să vadă și să atingă mașina pe care o folosesc. GCP oferă exemple pas cu pas de a face multe dintre cele mai frecvente sarcini: de exemplu, pornirea unei mașini virtuale bazate pe Linux, care este foarte simplu de făcut, utilizând ajutorul oferit de Google.

### 2.3.1 Firebase

Firebase este un set de instrumente pentru a construi, îmbunătăți și dezvolta aplicații, iar instrumentele oferite acoperă o mare parte din serviciile pe care dezvoltatorii ar trebui în mod normal să le construiască, dar nu o fac fiindcă prefera să se concentreze asupra aplicației în sine. Acest set include lucruri precum analize, autentificare, baze de date, configurare, stocare de fisiere, etc.

Firebase și Google Cloud partajează trei produse: Cloud Firestore, Cloud Functions și Cloud Storage. Acestea sunt aceleași produse care există în Google Cloud Platform, expuse pur și simplu pentru dezvoltatorii din partea clientului prin Firebase. Se pot accesa aceleași date din SDK-urile serverului (Google Cloud) și din SDK-urile clientului (Firebase), astfel încât echipele de frontend și backend să poată lucra împreună.

Firebase oferă două soluții de baze de date bazate pe cloud, accesibile pentru clienți, care acceptă sincronizarea datelor în timp real:

* Cloud Firestore: cea mai nouă bază de date Firebase pentru dezvoltarea aplicațiilor mobile. Se bazează pe succesul bazei de date în timp real cu un nou model de date mai intuitiv. Cloud Firestore oferă, de asemenea, interogări și scale mai bogate, mai rapide decât baza de date în timp real.
* Baza de date în timp real: baza de date originală a Firebase. Este o soluție eficientă, cu latență redusă pentru aplicațiile mobile care necesită stări sincronizate între clienți în timp real.

Alegerea soluției de bază de date depinde de mulți factori, dar exista unele recomandări bazate pe caracteristicile aplicației.

Ambele soluții oferă:

* SDK-uri primite de client, fără servere de implementat și întreținut
* Actualizări în timp real
* Initial este gratuit, apoi dezvoltatorul plătește ce folosește

Este bine de menționat faptul ca Realtime Database și Cloud Firestore

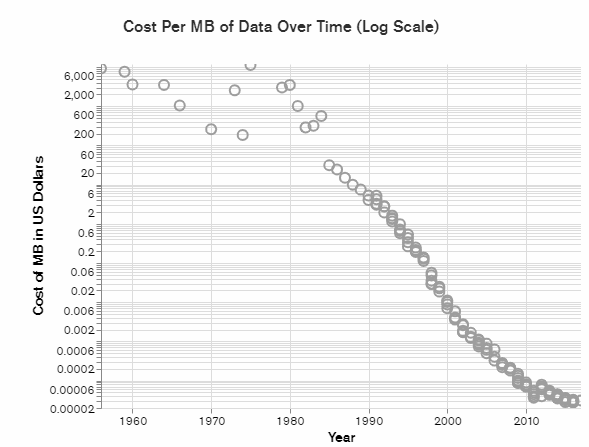
sunt baze de date de tipul NoSQL.

Bazele de date NoSQL (nu numai SQL) nu sunt tabulare și stochează date diferit de tabelele relaționale. Bazele de date NoSQL vin într-o varietate de tipuri pe baza modelului lor de date. Principalele tipuri sunt document, valoare cheie, coloană largă și grafic. Acestea oferă scheme flexibile și pot scala ușor cu cantități mari de date și încărcări mari ale utilizatorilor.

O concepție greșită obișnuită este că bazele de date NoSQL sau bazele de date non-relaționale nu stochează bine datele relației dintre tabele. Bazele de date NoSQL pot stoca date de relație - doar că le stochează diferit față de bazele de date relaționale. De fapt, în comparație cu bazele de date SQL, mulți consideră că datele relației de modelare din bazele de date NoSQL sunt mai ușoare decât în ​​bazele de date SQL, deoarece datele aferente nu trebuie împărțite între tabele.

Bazele de date NoSQL au apărut la sfârșitul anilor 2000, deoarece costul stocării a scăzut dramatic. Dezvoltatorii (în loc de stocare) deveneau costul principal al dezvoltării de software, astfel încât bazele de date NoSQL erau optimizate pentru productivitatea dezvoltatorilor.

Pe măsură ce costurile de stocare au scăzut rapid, a crescut cantitatea de aplicații de date necesare pentru stocare și interogare. Aceste date au apărut în toate formele și dimensiunile: structurate, semistructurate și polimorfe, și definirea schemei în avans a devenit aproape imposibilă. Bazele de date NoSQL permit dezvoltatorilor să stocheze cantități uriașe de date nestructurate, oferindu-le multă flexibilitate.



#### 2.3.1.1 Firestore

Cloud Firestore este o bază de date flexibilă și scalabilă pentru dezvoltarea de dispozitive mobile, web și de server de la Firebase și Google Cloud. La fel ca baza de date Firebase Realtime, vă păstrează datele sincronizate între aplicațiile clientului prin ascultători în timp real și oferă suport offline pentru mobil și web, astfel încât să puteți construi aplicații receptive care funcționează indiferent de latența rețelei sau de conectivitatea la Internet. Cloud Firestore oferă, de asemenea, o integrare perfectă cu alte produse Firebase și Google Cloud, inclusiv funcții Cloud.

Firestore este o bază de date NoSQL găzduită în cloud, pe care aplicațiile iOS, Android și web le pot accesa direct prin SDK-uri native. Este, de asemenea, disponibil în Node.js native, Java, Python, Unity, C ++ și SDK-uri Go, pe lângă API-urile REST și RPC. Urmând modelul de date NoSQL al Cloud Firestore, se stochează date în documente care conțin câmpuri de mapare a valorilor. Aceste documente sunt stocate în colecții, care sunt containere pentru documente care se pot utiliza pentru a organiza date și a crea interogări.

Documentele acceptă mai multe tipuri de date diferite, de la șiruri și numere simple, până la obiecte complexe. De asemenea, se pot crea subcolecții în documente și structuri ierarhice de date care se extind pe măsură ce baza de date crește. Modelul de date Cloud Firestore acceptă orice structură de date funcționează cel mai bine pentru dezvoltator.

În plus, interogarea în Cloud Firestore este expresivă, eficientă și flexibilă. Se pot crea interogări superficiale pentru a prelua date la nivelul documentului fără a fi nevoie să se preia întreaga colecție sau orice colecții imbricate. Se poate adăuga sortare, filtrare și limite la interogări sau cursoare pentru a pagina rezultatele. Pentru a menține actualizate datele din aplicații, fără a prelua întreaga bază de date de fiecare dată când are loc o actualizare, se pot adauga real-time listeners. Adăugarea de real-time listeners la o aplicație notifică dezvoltatorul ori de câte ori datele aplicațiilor clientului observă modificări, preluând doar noile modificări.

De asemenea, se poate proteja accesul la datele din Cloud Firestore cu Firebase Authentication și Cloud Firestore Security Rules pentru Android, iOS și JavaScript, sau Identity and Access Management (IAM) pentru limbajele de pe server.

#### 2.3.1.2 Hosting

Firebase Hosting oferă găzduire rapidă și sigură pentru aplicații web, conținut static și dinamic și microservicii. Aceasta este un tip de găzduire de conținut web de calitate pentru dezvoltatori. Cu o singură comandă, se poate implementa rapid aplicații web și se poate partaja atât conținut static cât și dinamic printr-un un CDN global (rețea de livrare a conținutului). De asemenea, Firebase Hosting se poate asocia cu funcții Cloud sau Cloud Run pentru a construi și găzdui microservicii pe Firebase.

Cele mai importante capabilități sunt:

* Servirea conținutului printr-o conexiune sigură. Web-ul modern este sigur. SSL de tip configurare zero este încorporat în Firebase Hosting, astfel încât conținutul este livrat întotdeauna în siguranță.
* Găzduire de conținut static și dinamic plus microservicii. Firebase Hosting acceptă toate tipurile de conținut pentru găzduire, de la fișierele CSS și HTML la microserviciile sau API-urile Express.js.
* Livrare rapida de continut. Fiecare fișier care este încărcat este stocat în cache pe SSD-uri pe marginile unui CDN din întreaga lume și este servit ca gzip sau Brotli. Se selectează automat cea mai bună metodă de compresie pentru conținut. Indiferent de unde sunt utilizatorii, conținutul este livrat rapid.
* Emularea/împărtășirea de modificări înainte de a lansa aplicația live. Se pot vizualiza și testa modificările pe o adresă URL găzduită local și se interacționează cu un backend emulat. Un dezvoltator își poate distribui modificările colegilor de echipă utilizând adrese URL de previzualizare temporare. Găzduirea oferă, de asemenea, o integrare GitHub pentru iterații ușoare ale conținutului.
* Implementarea de versiuni noi cu o singura comanda. Folosind CLI-ul Firebase, se poate pune aplicația în funcțiune în câteva secunde. Instrumentele din linia de comandă facilitează adăugarea targeturilor de implementare în procesul de construire. Și dacă trebuie să se anuleze implementarea, găzduirea oferă reveniri la versiuni anterioare cu un singur clic.

Firebase Hosting este conceput pentru dezvoltatorul web modern. Site-urile și aplicațiile sunt mai puternice ca oricând, odată cu creșterea frameworkurilor JavaScript front-end precum Angular, precum și instrumentele de generare statice, cum ar fi Jekyll. Fie că se implementează o pagină de destinație simplă a aplicației sau o aplicație web progresivă complexă (PWA), Găzduirea oferă infrastructura, caracteristicile și instrumentele adaptate implementării și gestionării site-urilor web și a aplicațiilor.

Pentru a servi continut, Firebase oferă mai multe opțiuni de domeniu și subdomeniu:

* În mod implicit, fiecare proiect Firebase are subdomenii gratuite pe domeniile web.app și firebaseapp.com. Aceste doua site-uri servesc același conținut și au același tip de configurație.
* Se pot crea mai multe site-uri dacă exista aplicații/site-uri conexe care servesc conținut diferit, dar partajează totuși aceleași resurse ale proiectului Firebase (de exemplu, dacă dezvoltatorul are un blog, un panou de administrare și o aplicație publica).
* Se poate conecta propriul domeniu la un site găzduit prin Firebase.

De asemenea, Firebase furnizează automat certificate SSL pentru toate domeniile, astfel încât tot conținutul să fie servit în siguranță.

Odată cu găzduirea gratuită și Cloud Firestore, Firebase oferă și niște reguli de securitate care se afla între datele dezvoltatorului și utilizatorii rău intentionati. Se pot scrie reguli simple sau complexe care protejează datele aplicației la orice nivel pe care aplicația specifică îl solicită.

Regulile de securitate Firebase utilizează limbaje de configurare extensibile și flexibile pentru a defini la ce date pot accesa utilizatorii aplicației din baza de date în timp real, Cloud Firestore și Cloud Storage. Regulile bazei de date în timp real Firebase folosesc JSON în definițiile regulilor, în timp ce regulile de securitate Cloud Firestore și regulile de securitate Firebase pentru stocarea în cloud utilizează un limbaj unic construit pentru a găzdui structuri mai complexe specifice regulilor.

Regulile de securitate Firebase funcționează potrivind un model cu căile bazei de date și apoi aplicând condiții personalizate pentru a permite accesul la date pe acele căi. Toate regulile produselor Firebase au o componentă de potrivire a căii și o declarație condițională care permite accesul la citire sau scriere. Trebuie să fie definite reguli pentru fiecare produs Firebase pe care dezvoltatorul îl utilizează în aplicație.

#### 2.3.1.3 Firebase CLI

Linia de comandă Firebase poate fi utilizată pentru a gestiona și implementa proiecte Firebase din linia de comandă.

* Poate servi cod către proiecte firebase
* Rulează un server web local pentru site-ul unde este folosit Firebase
* Poate interactiona cu datele din baza de date Firebase
* Importă/exportă utilizatori din Firebase Auth

Presupunand ca exista un folder cu un proiect Vue numit proiect, aceștia sunt pașii folosiți pentru a integra Firebase:

* Trebuie creat un proiect pe site-ul Firebase
* Se selectează pictograma Web
* Se face click pe tab-ul de Hosting de pe site și se selectează locația unde dezvoltatorul dorește să își găzduiască aplicația
* Se face click pe tab-ul de Database, și se selectează modul de test
* Odată cu aceste configurații, dezvoltatorul o să primească informații folositoare pentru a configura Firebase în proiect, sub forma unui fisier JSON. Acestea ar trebui importate sau puse în fișierul main.js
* Se instalează Firebase din linia de comandă:



* După instalare, dezvoltatorul trebuie să se autentifice:



* Pentru a verifica dacă totul a fost instalat corect, se pot lista proiectele Firebase:



* Se inițializează proiectul Firebase. Multe activități obișnuite efectuate utilizând CLI, cum ar fi implementarea într-un proiect Firebase, necesită un director de proiect. Trebuie stabilit un director de proiect utilizând comanda firebase init. Un director de proiect este, de obicei, același director cu rădăcina de control sursă și, după executarea firebase init, directorul conține un fișier de configurare firebase.json.

Comanda firebase init ghidează dezvoltatorul prin configurarea directorului de proiect și a unor produse Firebase. În timpul inițializării proiectului, Firebase CLI solicită completarea următoarelor sarcini:

* Selectarea produselor Firebase dorite pentru a fi configurate în proiect. Acest pas solicită setarea unor configurații pentru fișiere specifice pentru produsele selectate. Acest pas poate fi rulat oricand pentru a configura mai multe produse.
* Selectarea unui proiect Firebase implicit. Acest pas asociază directorul curent al proiectului cu un proiect Firebase, astfel incat comenzile specifice proiectului (cum ar fi implementarea Firebase) să ruleze pe proiectul Firebase corespunzător.
* La sfârsitul inițializării, Firebase creează automat următoarele doua fișiere în rădăcina proiectului:
  + firebase.json: un fișier de configurare care listează configurația proiectului.
  + .firebaserc: un fișier care stochează aliasurile proiectului.