Proiect "Nikola"

- concurs: InfoEducatie, editia 2017
- categorie: Soft educational
- homepage: http://178.62.96.120/

I. Introducere

Nikola este un proiect educational de tip "website", vizand categoria concursului "Soft educational". A fost dezvoltat cu scopul de a ajuta elevii si profesorii ce predau informatica, plecand de la urmatoarele cerinte:

- elevii sa se poata loga fara a tine minte alt cont
- elevii sa aiba access la o platforma portabila, in care sa poata creea programe, cat si sa rezolve probleme
- elevilor sa le fie propuse metode eficiente de rezolvare ale unor probleme
- elevii sa nu depaseasca o limita de timp, propusa de profesori
- profesorii sa poata gestiona rapid datele aflate pe platforma
- profesorii sa poata modifica oricand problema propusa

Astfel, echipa a dezvoltat platforma Nikola, care sa ajute la imbunatatirea invatamantului romanesc si al invatarii individuale.

Numele platformei, Nikola, este un omagiu adus inventatorului si fizicianului sarb-american Nikola Tesla, care a contribuit definitiv in dezvoltarea tehnologiei.

II. Utilizabilitate, originalitate si inovatie

Nikola poate fi accesata ca website, utilizand adresa din incipitul documentatiei, dar si aplicatie de desktop atasata proiectului.

Odata accesata platforma, elevul este intampinat cu o pagina de start, in care este invitat sa se logheze cu contul sa de Github, platforma foarte populara pentru tinerii pasionati de IT. Dupa confirmarea logarii, el este redirectionat catre dashboard.

Dashboard-ul reprezinta aplicatia propriu-zisa. In partea stanga a aplicatiei se afla un meniu in care se afla butoane cu diferite actiuni:

- "Compileaza", cu un buton atasat, care, odata actionat, va compila programul introdus de user, folosind-use de datele de intrare, si va returna datele de iesire si informatii despre compilarea programului
- "Problema actuala", care, actionat, va actualia recomandarile profesorilor, aflate in dialog box-ul din coltul sus-dreapta, si va afisa in fereasta de mai jos cerinta problemei si format-ul datelor de intrare si de iesire; langa acesta apare un buton care va trimite datele de iesire returnare de program si le va compara cu cele corecte, introduse de profesori
- "Informatii", care, odata actionat, va afisa datele preluate de pe contul personal de Github al utilizatorului; langa acesta apare un buton folosite pentru delogare

Compilarea se poate realiza in mai multe limbaje de programare, a caror selectare poate fi motivata astfel:

- C: limbaj primordial
- C#: limbaj specific aplicatiilor Windows
- C++: limbaj predat in scolile romanesti
- Java: limbaj specific aplicatiilor mobile Android
- Javascript: limbaj specific aplicatiilor web
- Pascal: limbaj predat in scolile romanesti
- Perl: limbaj de script-ing
- PHP: limbaj folosit pentru serverele aplicatiilor web
- Python: limbaj foarte popular la nivel global
- Swift: limbaj specific aplicatiilor mobile IOS

In continuarea acestor butoane, se afla o fereasta folosita pe post de frame, in care se vor afisa diverse informatii. Un form de selectare permite utilizatorilor sa selecteze limbajul de programare al programelor. Widget-urile din partea de jos a meniului afiseaza:

- ora actuala, actualizata permanent
- vremea de afara, afisata deoarece promovam si iesirile periodice afara, in lipsa tehnologiei
- numarul de minute si ore petrecute pe platforma

- recomandarea platformei, prin care elevul este motivat sa fie productiv sau anuntat sa ia o pauza, in cazul in care a trecut o ora de cand se afla pe website

In coltul sus-dreapta al paginii este prezent un dialog box, in care este afisata recomandarea profesorului cu privire la rezolvarea problemei. Restul paginii este alcatuit din 4 editoare de text:

- editorul, in care se va introduce programul; acesta prezinta o evidentiere a sintaxei programului, specifica limbajului selectat, cat si asa-numitii "workeri", care afiseaza in timp real erorile, prezenti insa pentru un numar redus de limbaje de programare
- informatii de compilare, incare sunt afisare detalii despre compilarea programului, precum: status-ul, result-ul, timpul de compilare, spatiul ocupat si erorile
- datele de intrare
- datele de iesire, respectiv rezultatul trimiterii problemei

Pagina de admin este disponibila doar profesorilor, care pot edita in timp real informatiile de pe paginile platformei, cat si problema propusa anterior. In partea de sus a paginii sunt prezente butoane cu care se poat accesa mai usor paginile platformei. Sub acestea, apar input-uri de tip text si butoane pentru actualizare.

Din punct de vedere al inovatiei si al originalitatii, platforma este unica prin posibilitatea compilarii de pe orice dispozitiv avand instalat un browser sau de pe un sistem compatibil cu aplicatia de desktop. Website-ul satisface un numar relativ mare din nevoile elevilor si al profesorilor. Logarea cu contul de Github nu impovareaza elevul cu alt cont specific platformei Nikola, ci, mai degraba, favorizeaza si folosirea website-ului Github, necesar unui viitor dezvoltator in domeniu.

Publicul-tinta este reprezentat de catre elevii care doresc sa invete programarea intr-un program si sa renunte la metodele clasice folosite in scolile romanest.

III. Interfata si continut

Nikola poseda o interfata intuitiva, usor de utilizat si adaptiva pe majoritatea browserelor si sistemelor de operare. Aspectul este placut, prin utilizatrea unei compinatii de albastru, alb si negru. Au fost folosite icon-uri specifice pentru butoane si efecte de hover, care confera o nuanta de modernitate interfetei.

Continutul platformei este corect din punct de vedere gramatical si stiintific. El poate fi modificat in timp real de catre profesori cu ajutorul paginii de admin.

IV. Arhitectura si programare

Pentru a realiza proiectul, s-au utilizat tehnologiile: CSS, Javascript, Node.js si MongoDB. Structura folder-elor este urmatoarea:

- desktop: contine aplicatia de desktop si proiectul acesteia
- node_modules: contine modulele NPM
- other/Database: contine o copie a bazei de date NoSQL
- other/Documentatie: contine documentatia de fata
- other/Screenshots: contine screenshot-uri ale aplicatiei
- other/Test: contine teste pentru compilator
- other/Video: contine videoclipul de prezentare
- server/config: contine datele de configurare ale server-ului Node.js
- server/controllers: contine controalele rutelor MVC
- server/models: contine modelele rutelor MVC
- server/my_modules: contine modulele create pentru a utiliza dezvoltarea
- server/public/lib: contine librariile .css sau .js utilizate pe platforma
- server/public/css: contine fisierele .css generate cu ajutorul Gulp.js
- server/public/img: contine imaginile utilizate pe platforma
- server/public/js: contine script-urile Javascript de pe partea de frontend
- server/public/less: contine fisierele .less folosite pentru a genera fisierele .css
- server/public/css: contine fisierele .css generate cu ajutorul Gulp.js
- server/public/projects/Favicon: contine proiectul favicon-ului

- server/public/projects/Github: contine proiectul logo-ului folosit pe aplicatia de Github
- server/public/views: contine fisierele .pug folosite pentru a genera fisierele .html, views pentru rutele MVC
- server/sockets: contine configurarea WebSockets-urilor
- .git: contine fisierele specifice repository-ului de Github

Logica server-ului se afla in app.js. In interiorul acestuia, sunt luate configuratiile din fisierul Javascipt config/config.json, de la port la contul de admin. Sunt preluate modulele NPM in prima parte a fisierului. Cu ajutorul mongoose, se creeaza o legatura cu baza de date MongoDB si modelele necesare preluarii datelor. In dezvoltarea server-ului, s-a folosit Express, pe care au fost setate diferite middlewares precum cors si hemplet. Pug este folosit ca template engine pentru a usura scrierea HTML. Pentru a stabili o conexiune constanta cu user-ul, am folosit socket.io.

Am folosit trei API-uri:

- Cloud Compiler API: compilarea programelor utilizatorului; la dezvoltarea acestuia am contribuit, impreuna cu un numar restrans de developeri indieni
- IP-API: preluarea locatiei in functie de IP-ul utilizatorului
- Open Weather Map: preluarea datelor meteorologice in functie de locatie

La un request, user-ul primeste un fisier .html, care reprezinta de fapt un fisier .pug, in care au fost introduse datele preluate cu ajutorul mongoose. Fisierele .css sunt compilate din fisierele .less, tehnologie folosita pentru a usura scrierea codului CSS. Ca framework de client side, am folosit Vue.js.

S-au folosit urmatoarele module instalate cu NPM:

- body-parser: modul folosit pentru a parsa body-urile request-urilor inainte de handlere
- browser-sync: modul folosit in Gulp.js pentru live reload
- cookie-parser: modul folosit pentru a parsa cookie-urile
- cors: modul folosit pentru a activa CORS
- electron: modul folosit pentru a realiza aplciatia de desktop
- express: modul folosit ca framework pe server-ul Node.js
- express-partials: modul folosit in Express pentru partials
- express-session: modul folosit pentru a seta sessions pe server
- gulp: modul de gestionare a task-urilor
- gulp-less: modul folosit pentru a crea un task pentru a compila fisierele .less
- helmet: modul folosit pentru a securiza aplicatia Node.js
- less: modul folosit pentru a compila fisierele .less
- method-override: modul pentru a activa verbele HTTP, precum PUT si DELETE, unde clientul nu suporta
- mongoose: modul pentru modelarea obiectelor din MongoDB
- passport: modul pentru a autentificarea in Node.js
- passport-github2: modul pentru autentificarea cu Github
- pug: modul folosit ca template engine, compliland fisierele .pug in .html
- request: modul folosit pentru a faca request-uri in Node.js
- socket.io: modul pentru a utiliza WebSockets pe partea de server

In fisierul gulpfile.js sunt precizate task-urile Gulp.js. Acest modul a fost esential in dezvoltarea aplicatiei deoarece am automatizat munca prin live reload si compilarea fisierelor less. Tot pentru dezvoltare, am folosit editorul Atom, browserele Crome si Firefox, GUI-ul pentru mongo RoboMongo, iar pentru versionare Github, asigurandu-se caracterul de open-source al platformei.

Codul sursa al aplicatiei este foarte bine structurat si este documentat cu ajutorul comentariilor, cu scopul de a putea fi inteles de posibili colaboratori. Design-ul este responsive, insa pe dispozitivele mobile este recomandata utilizarea landscape. Testarea a fost realizata pe browserele Chrome, Firefox si pe sistemele de operare Windows si Ubuntu. Hostarea aplicatiei s-a realizat cu ajutorul Digital Cloud. Pentru asignarea aplicatiei pe root-ul web-server-ului, am folosit nginx, astfel se evita afisarea portului pe care aplciatia ruleaza si se asigura inca un strat de securitate.

Securitatea aplicatiei este asigurata prin modulul helmet, prin protejarea aplicatie de majoritatea atacuri specifice Node.js si prin verificarea datelor utilizatorului.

V. Dezvoltare ulterioara

In viitor, aplicatia va fi prezentata in mod oficial unor profesori si elevi, pentru a primi un feedback dupa care echipa sa isi dea seama daca aplicatia starneste interes si daca ar trebui sa continue dezvoltarea, caz in care se pot aduce urmatoarele imbunatatiri:

- adaugarea unei arhive cu problemele periodice
- posibilitatea elevilor de a-si putea customiza lista de limbaje de programare
- adaugarea a mai multe metode de login cu ajutorul retelelor de socialziare

VI. Ghid de instalare si de configurare

Platforma Nikola poate fi testata prin accesarea website-ului(link-ul este atasat la inceputul documentatiei) sau local cu ajutorul Node.js si a MongoDB, urmand pasii:

- instalare Node.js
- instalare si rulare MongoDB
- instalare RoboMongo
- importare in Robomongo a bazei de date NoSQL, aflata in folder-ul /database
- instalare dependente proiect cu ajutorul NPM
- configurarea variabilelor specifice cu ajutorul fisierului config.json aflat in folder-ul /config
- rulare app.js cu ajutorul Node.js

Instalarea RoboMongo este optionala in cazul in care se va folosi sheel-ul mongo, instalat odata cu MongoDB. In cazul in care intampinati probleme cu instalarea modulelor de pe NPM, va invit sa clonati repository-ul de pe Github: https://github.com/iosifache/Nikola.git.

Pentru a avea access si la panoul de admin, se va folosi pentru logare email-ul "nikolaforpreview@outlook.com" si parola "empowersoft2017".

VII. Opinie

Justificarea tehnologiilor alese: am ales Node.js ca runtime environment deoarece acesta prezinta numeroase avantaje. Fiind un proiect relativ tanar fata de celelate alternative folosite pentru a creea un server, Node.js este bazat de Javascript, folosindu-se astfel acelasi limbaj de programare pentru frontend si pentru backend. Folosind engine-ul V8 al celor de la Google si asynchronous I/O, viteza cu care ruleaza este foarte mare. Dispune de un manager de module, numit NPM, si este recomandat sa se foloseasca cu acest tip de aplicatii baze de date NoSQL, implementate si in aplicatia de fata cu ajutorul MongoDB.

Opinia despre ideea de baza a proiectului si despre utilitate: suntem toti consitenti ca invatamantul din Romania nu este cel mai eficient, in comparatie cu sistemele din strainatate. Astfel, fiind contienti ca o simpla aplicatie web, realizata in mai putin de jumatate de luna, nu va revolutiona tot sistemul de invatamant, am incercat sa imbunatatim predarea si invatarea unei materii: informarica. Aceasta materie reprezinta, din punctul nostru de vedere, temelia generatiilor urmatoare care vor trai intr-un viitor mult mai tehnologizat. Prin utilitatea aplicatiei, ce dorim a fi folosita in scolile romanesti, vrem sa starnim interesul elevilor pentru Informatica, cu scopul final de a avea in viitor, poate in urmatorii 5-10 ani tineri mult mai pregatiti in domeniul IT.