

# Vue CLI

U prethodnoj lekciji smo se zaustavili kod jedne od najmoćnijih funkcionalnosti Vuea, koja omogućava definisanje komponentata unutar jednog fajla. Takve komponente se nazivaju Single-file komponente, a njihov kôd se smešta unutar fajlova sa ekstenzijom `.vue`. Na taj način Vue u pravom smislu te reči omogućava da se izgradnja korisničkog okruženja jedne aplikacije obavi potpuno modularno, izolovanjem pojedinačnih sekcija u zasebne fajlove.

Ono što ste u prethodnoj lekciji mogli da pročitate odnosi se na osobenost isporuke Vue aplikacija koje se kreiraju korišćenjem Single-file komponentata. Naime, web pregledači nisu u stanju da direktno razumeju sadržaj fajlova sa `.vue` ekstenzijom, pa je tako u priču o Vue razvoju neophodno uključiti još neke pojmove. Pre svih, misli se na komandni interfejs Vue CLI, koji se može koristiti da olakša i ubrza Vue razvoj. Takav alat, između ostalog, omogućava i da se `.vue` fajlovi transformišu u zasebne fajlove sa HTML, CSS i JavaScript kodom. Stoga će lekcija pred vama u potpunosti biti posvećena instalaciji i korišćenju Vue CLI alata.

## Šta je Vue CLI?

Vue CLI (engl. *Vue Command Line Interface*) je alat koji je Vue tim razvio kako bi olakšao razvoj Vue aplikacija. Reč je zapravo o skupu funkcionalnosti koje se na razvojni računar instaliraju kao globalni npm paket. To praktično znači da je reč o funkcionalnostima koje za svoje izvršavanje zahtevaju postojanje izvršnog okruženja Node.js. Tako prvi put dolazimo do situacije koja je spomenuta u jednoj od prethodnih lekcija, a tiče se činjenice da su Node.js i npm nezaobilazne komponente modernog frontend programiranja.

Osnovne prednosti koje Vue CLI donosi su:

- lako kreiranje novih projekata i komponentata,
- mogućnost živog testiranja koda, što podrazumeva automatsko propagiranje promena iz izvornog koda do web pregledača,
- mogućnost lake integracije brojnih dodatnih funkcionalnosti (modula, odnosno paketa),
- laka izgradnja produkcionog koda, koji je spreman za to da ga izvršavaju web pregledači.

## Struktura Vue CLI sistema

Vue CLI sačinjen je iz tri osnovna dela:

- CLI,
- CLI Service,
- CLI Plugins.

**CLI** je npm paket koji se globalno instalira na razvojnom kompjuteru. Reč je o npm paketu koji nosi naziv **@vue/cli** (to je naziv po kome ovaj paket možete pronaći unutar npm repozitorijuma). CLI obezbeđuje korišćenje `vue` komande unutar konzole ili terminala.

Paralelu možete napraviti sa već viđenim komandama, kao što su `node` ili `npm`. `node` je osnovna komanda pomoću koje je moguće upućivati naredbe izvršnom okruženju `Node.js`, dok je `npm` osnovna komanda `npm` menadžera paketa. Po istoj logici funkcioniše i komanda `vue`.

Druga značajna celina Vue CLI ekosistema je **CLI Service**. Opet je reč o jednom `npm` paketu, ali ovoga puta o paketu koji se instalira lokalno, odnosno zasebno, unutar svakog projekta koji se kreira korišćenjem Vue CLI-ja. Pun naziv `npm` paketa koji predstavlja CLI Service je **@vue/cli-service**.

CLI Service obavlja sledeće, vrlo važne operacije prilikom korišćenja Vue CLI-ja:

- obezbeđuje tri dodatne komande – `serve`, `build` i `inspect`,
- omogućava učitavanje dodatnih `npm` paketa koji će biti korišćeni unutar projekta,
- obavlja automatsko konfigurisanje webpacka na osnovu osobina Vue aplikacije.

Na kraju, prilikom razvoja Vue aplikacija moguće je koristiti i brojne `npm` pakete, koji se unutar Vue sveta nazivaju CLI Plugini. **CLI Plugini** obezbeđuju korišćenje različitih dodatnih alata i funkcionalnosti prilikom razvoja Vue aplikacija. Nazivi svih takvih `npm` paketa započinju prefiksom `@vue/cli-plugin-` ili `vue/cli-plugin-`.

#### Šta predstavlja karakter @ na početku naziva npm paketa?

U prethodnim redovima ste mogli da vidite da nazivi konkretnih `npm` paketa koji se koriste prilikom Vue razvoja započinju karakterom `@` nakon koga se definiše naziv `vue` (`@vue`). Unutar `npm` repozitorijuma, karakter `@` se koristi kao prefiks za definisanje naziva organizacije kojoj takvi paketi pripadaju. Stoga, prefiks `@` ispred naziva `vue` znači da je reč o `npm` modulu koji je kreirala organizacija `vue`. Paketi (moduli) čiji naziv započinje karakterom `@` drugačije se nazivaju *Scoped packages* ili *Scoped modules*, zato što omogućavaju da se veći broj `npm` paketa efikasno podvede pod jednu oblast, odnosno prostor imena.

Korišćenje karaktera `@` omogućava da se na veoma lak način distanciraju zvanični od nezvaničnih paketa (modula). Oni paketi koji ne poseduju karakter `@` nisu zvanični i obrnuto.

#### Pitanje

S obzirom na to da se izvršava posredstvom `Node.js`-a, kojim jezikom je napisan Vue CLI?

- a) PHP jezikom
- b) JavaScript jezikom**
- c) Java jezikom
- d) HTML jezikom

#### Objašnjenje:

*Vue CLI je skup funkcionalnosti koje se na razvojni računar instaliraju kao globalni npm paket. Sasvim logično, takav paket je napisan JavaScript jezikom, a izvršava se posredstvom izvršnog okruženja Node.js.*

## Instalacija Vue CLI-ja

Pre nego što budemo u mogućnosti da jedan projekat kreiramo korišćenjem Vue CLI-ja, neophodno je obaviti njegovu instalaciju kao globalnog npm paketa. Stoga je preduslov da na vašem kompjuteru postoji instaliran Node.js, zajedno sa npm menadžerom paketa. U jednoj od prethodnih lekcija je ilustrovana kompletna procedura za obavljanje takvog posla. Kako biste se uverili da na vašem kompjuteru postoje Node.js i npm, moguće je iskoristiti komande za proveru njihovih verzija. Stoga, za proveru raspoloživosti Node.js-a unutar konzole ili terminala možete da unesete sledeće:

```
node -v
```

Rezultat izvršavanja ovakve komande treba da bude verzija izvršnog okruženja Node.js koja je instalirana na sistemu.

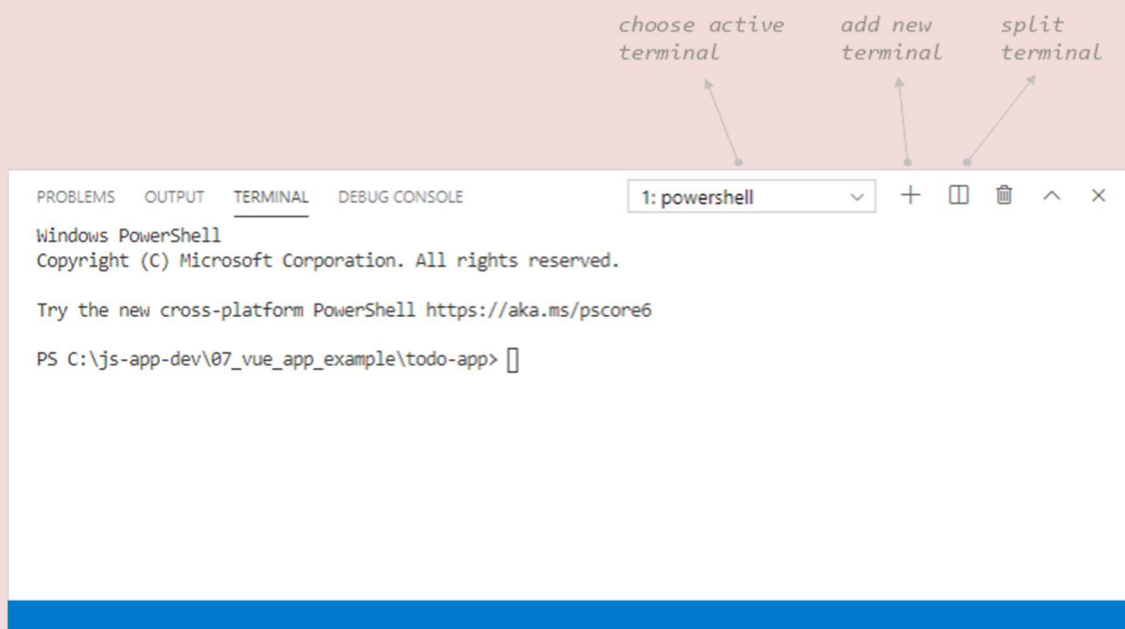
Provera raspoloživosti npm menadžera paketa može se obaviti upućivanjem sledeće komande:

```
npm -v
```

I sada je kao rezultat ovakve komande potrebno dobiti verziju instaliranog npm menadžera paketa.

### Terminal unutar Visual Studio Codea

Prikazane komande namenjene izvršavanju od konzole ili terminala na vrlo jednostavan način moguće je upućivati korišćenjem Visual Studio Code tekst editora. Naime, sastavni deo Visual Studio Codea jeste i Terminal koji omogućava olakšano korišćenje izvorne konzole ili terminala u zavisnosti od operativnog sistema na kome je VS Code instaliran.



Slika 7.1. VS Code Terminal

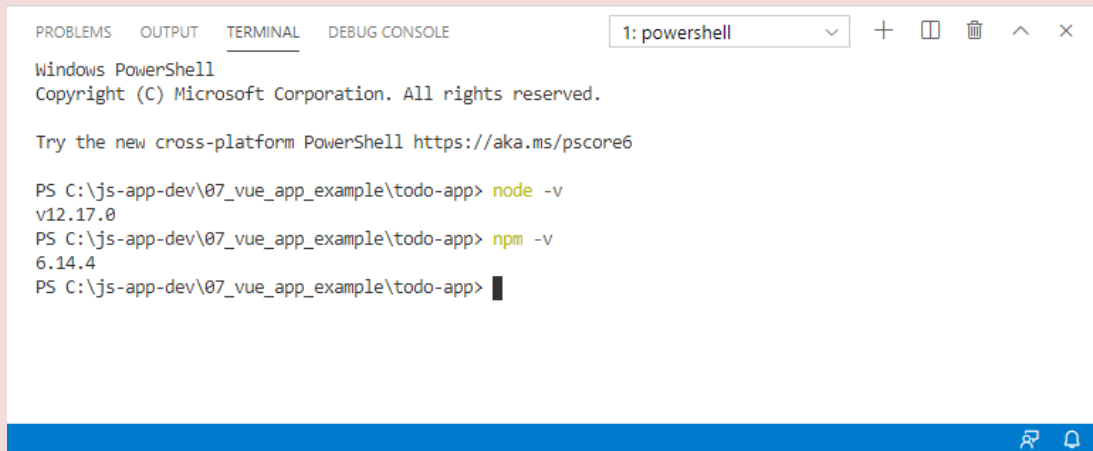
Terminal prikazan slikom 7.1. može se otvoriti na nekoliko načina:

- odabirom opcije **Open in Terminal** iz kontekstnog menija, koji se dobija desnim klikom na bilo koji fajl ili folder,
- odabirom opcije **View -> Terminal**,
- odabirom opcije **Terminal -> New Terminal**.

Pored mogućnosti lakog korišćenja konzole ili terminala, Visual Studio Code Terminal obezbeđuje i brojne napredne funkcionalnosti:

- putanja unutar terminala se uvek postavlja na lokaciju trenutnog dokumenta, stoga nije potrebno gubiti vreme kako bi se obavilo pozicioniranje unutar foldera projekta,
- moguće je otvoriti veći broj terminala klikom na plus (+) dugme unutar zaglavlja terminala,
- trenutno vidljiv terminal je moguće odabrati iz padajućeg menija pored dugmeta za dodavanje,
- moguće je i u jednom trenutku otvoriti veći broj terminala u režimu podeljenog prikaza (opcija Split Terminal).

Sve ovo znači da je komande za proveru raspoloživosti Node.js-a i npm-a moguće uputiti na način prikazan slikom 7.2.

The image shows a screenshot of the Visual Studio Code interface with the 'TERMINAL' tab selected. The terminal is running a Windows PowerShell session. The title bar of the terminal window shows '1: powershell'. The output in the terminal includes the standard PowerShell copyright notice and a message about the new cross-platform PowerShell. Below this, three commands are entered: 'node -v' which outputs 'v12.17.0', 'npm -v' which outputs '6.14.4', and a third command 'PS C:\js-app-dev\07\_vue\_app\_example\todo-app>' with a cursor. The bottom status bar of the terminal window is blue and contains icons for refreshing and muting.

*Slika 7.2. Korišćenje VS Code Terminala*

Nakon osiguravanja postojanja izvršnog okruženja Node.js i npm menadžera paketa, moguće je preći na instalaciju Vue CLI-ja. To se postiže upućivanjem sledeće komande:

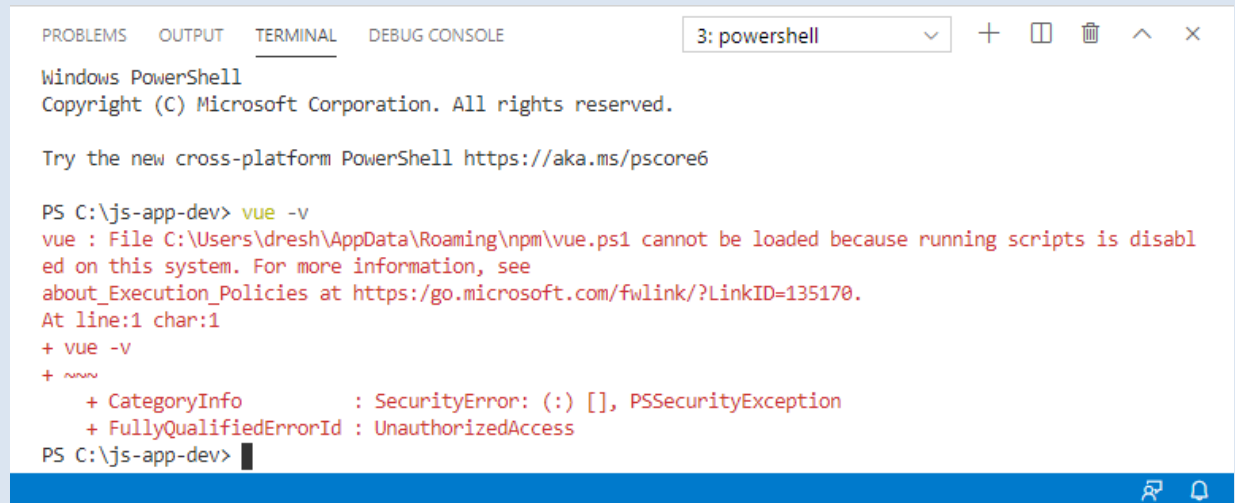
```
npm install -g @vue/cli
```

Na ovaj način obavlja se globalna instalacija jednog npm paketa, odnosno Node.js modula. Izvršavanje komande će trajati neko vreme, stoga pričekajte da se instalacija Vue CLI-ja završi, a nakon toga uspešnost instalacije proverite na sledeći način:

```
vue -V
```

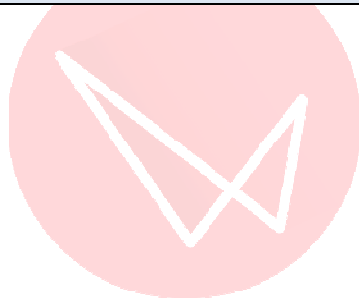
## Napomena

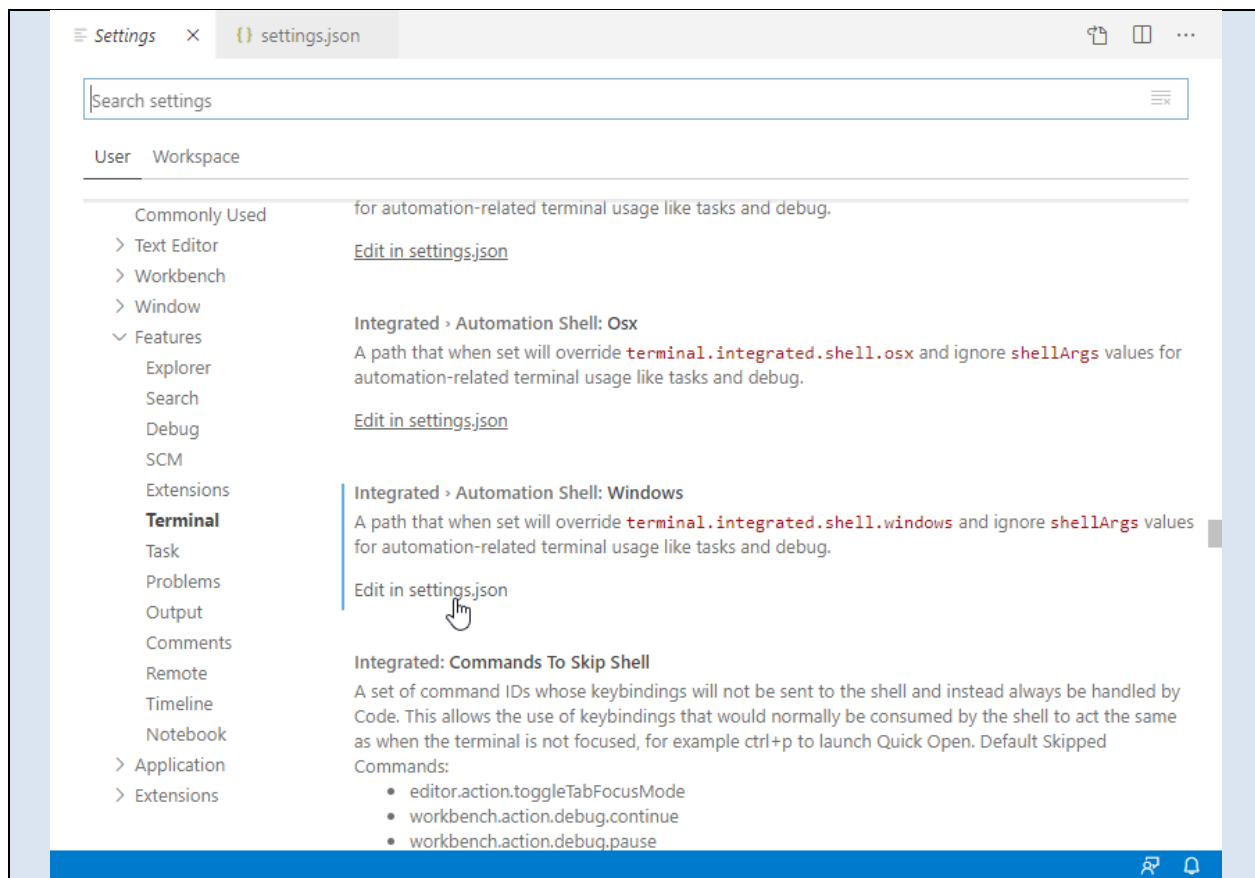
Može se dogoditi da prilikom upućivanja upravo prikazane komande korišćenjem VS Code Terminala na operativnom sistemu Windows, dobijete grešku kao na slici 7.3.

The image is a screenshot of the VS Code interface, specifically the Terminal panel. The terminal is running a Windows PowerShell session. The prompt is 'PS C:\js-app-dev>'. The user has entered the command 'vue -v'. The output shows an error: 'File C:\Users\dresh\AppData\Roaming\npm\vue.ps1 cannot be loaded because running scripts is disabled on this system. For more information, see about\_Execution\_Policies at https://go.microsoft.com/fwlink/?LinkID=135170. At line:1 char:1 + vue -v + ~~~~~ + CategoryInfo : SecurityError: (:) [], PSSecurityException + FullyQualifiedErrorId : UnauthorizedAccess'. The terminal window has a title bar that says '3: powershell'. The background of the terminal is dark blue, and the text is white and red. The error message is in red. The prompt is in white. The user's input is in white. The output is in white and red. The terminal window is part of a larger VS Code interface, which is visible in the background. The top of the interface shows the 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE' tabs. The 'TERMINAL' tab is selected. The bottom of the interface shows a blue status bar with icons for search, run and debug, and a bell icon.

*Slika 7.3. Greška do koje može doći prilikom korišćenja VS Code Terminala na operativnom sistemu Windows*

Razlog zbog koga se može dobiti ovakva greška jeste postojanje sigurnosnih polisa unutar aplikacije PowerShell, koju na operativnim sistemima Windows VS Code koristi za realizaciju Terminala. Kada se PowerShell integriše unutar nekog drugog sistema, odnosno aplikacije (kao što je to slučaj unutar VS Codea), neophodno je ručno definisati politiku koja će omogućiti da se ovakve komande nesmetano upućuju. Sve što je potrebno da uradite jeste da odaberete **File -> Preferences -> Settings**, a zatim iz spiska podešavanja sa leve strane **Features -> Terminal** (slika 7.4).





*Slika 7.4. Opcija za konfigurisanje VS Code Terminala na operativnom sistemu Windows*

Potrebno je pronaći odeljak koji omogućava podešavanje Terminala za operativne sisteme Windows i kliknuti na **Edit in settings.json**, baš kao na slici 7.4. U JSON fajl, koji se na ovaj način otvara, potrebno je dodati sledeći par ključeva i vrednosti:

```
"terminal.integrated.shellArgs.windows": ["-ExecutionPolicy", "Bypass"]
```

## Kreiranje novog projekta korišćenjem Vue CLI-ja

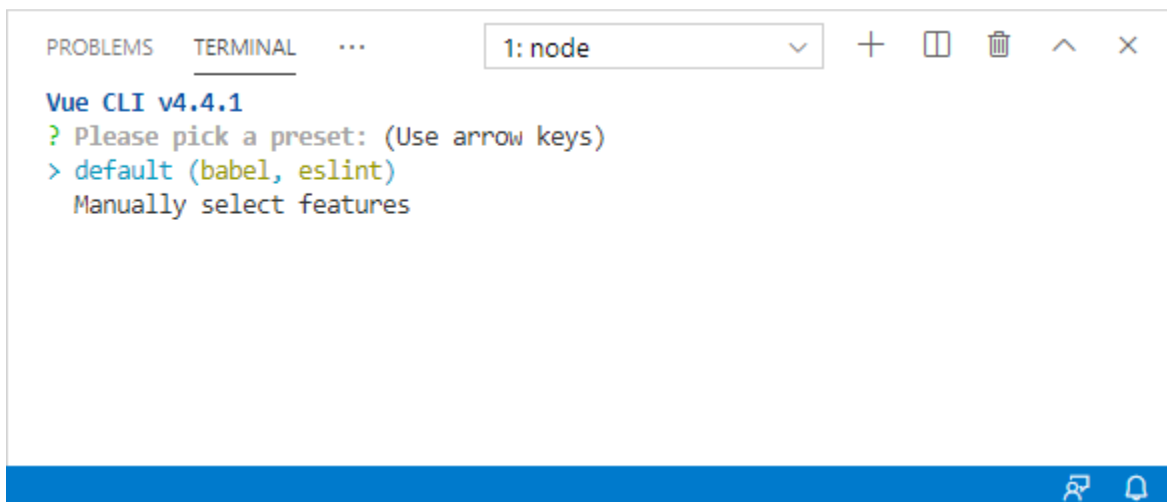
Nakon instalacije Vue CLI-ja, moguće je prvi put kreirati jedan nov projekat korišćenjem takvog alata. Postoje dva načina za obavljanje takvog posla:

- korišćenje komandnog interfejsa i
- korišćenje grafičkog interfejsa.

Mi ćemo primarno koristiti komandni interfejs, što podrazumeva upućivanje komandi upravo instaliranom Vue CLI-ju korišćenjem konzole ili terminala. Kreiranje novog projekta na taj način obavlja se korišćenjem komande `vue create`:

```
vue create todo-app
```

Upućivanjem ovakve komande započinje procedura kreiranja Vue projekta (slika 7.5).



```
PROBLEMS  TERMINAL  ...  1: node  +  [icon]  [icon]  ^  x

Vue CLI v4.4.1
? Please pick a preset: (Use arrow keys)
> default (babel, eslint)
   Manually select features
```

Slika 7.5. Odabir šablona na osnovu koga će biti kreiran Vue projekat

Vue CLI omogućava da se prilikom kreiranja projekta odabere i skup funkcionalnosti koje će biti korišćene unutar projekta. Naime, Vue CLI programeru na raspolaganje stavlja nekoliko veoma korisnih biblioteka i alata koji mogu pozitivno uticati na produktivnost i brzinu kreiranja projekta. Mi ćemo se odlučiti za podrazumevani skup funkcionalnosti koji zahteva korišćenje biblioteka *Babel* i *ESLint*.

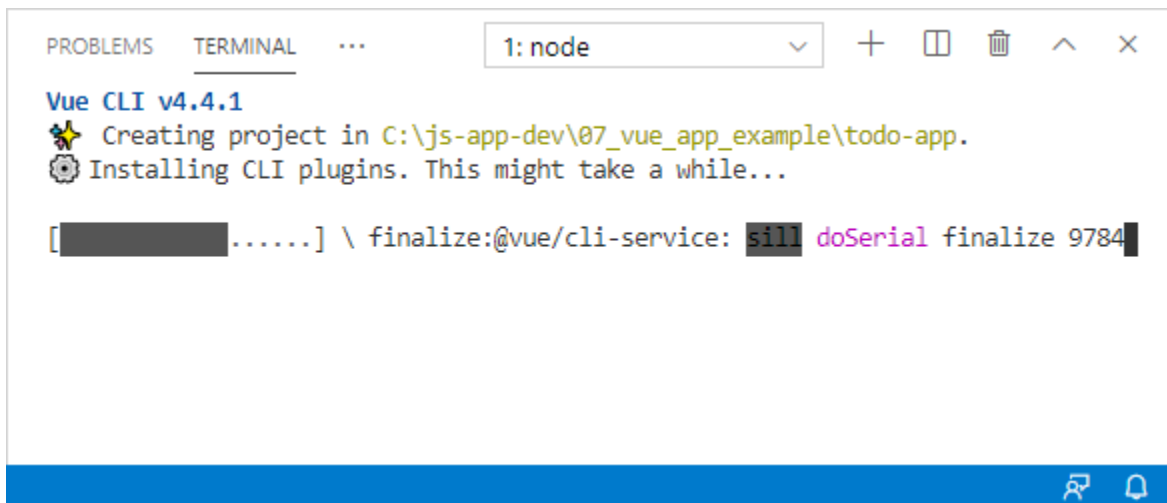
### Babel

Babel je JavaScript biblioteka koja je primarno namenjena konvertovanju JavaScript koda koji je napisan uz poštovanje novijih verzija ECMAScript specifikacije u kod koji mogu izvršiti i web pregledači i izvršnih okruženja koja takve nove funkcionalnosti ne podržavaju. Takvi alati, koji obavljaju pretvaranje programskog koda jednog jezika u optimizovani kod istog takvog jezika, drugačije se nazivaju transpajleri (engl. *transpiler*), pa je tako Babel – alat za transpajliranje. S obzirom na to da je sa pojavom verzije ES6 (ECMAScript 2015) uvedeno dosta novih jezičkih funkcionalnosti, Babel se prevashodno koristi za prevođenje koda koji se zasniva na ES6 i novijim specifikacijama.

### ESLint

ESLint je biblioteka koja se bavi analiziranjem JavaScript koda zarad pronalaska grešaka ili problematičnih delova koji ne zadovoljavaju unapred definisan standard. Pored toga, ESLint omogućava i lako formatiranje koda. Stoga je reč o biblioteci koja obezbeđuje funkcionalnosti po ugledu na mnoge moderne tekst editore i razvojna okruženja.

Za odabir podrazumevanog skupa funkcionalnosti dovoljno je da pritisnete *Enter*, bez bilo kakvog unosa, i proces kreiranja projekta će da započne (slika 7.6).



```
PROBLEMS  TERMINAL  ...  1: node  +  [icon]  [icon]  ^  x

Vue CLI v4.4.1
✦ Creating project in C:\js-app-dev\07_vue_app_example\todo-app.
⚙ Installing CLI plugins. This might take a while...

[REDACTED] .....] \ finalize:@vue/cli-service: sill doSerial finalize 9784
```

Slika 7.6. Kreiranje Vue projekta korišćenjem Vue CLI-ja

### Grafičko korisničko okruženje za rukovanje Vue CLI-jem

Nešto ranije je rečeno da Vue CLI poseduje i grafičko korisničko okruženje koje je moguće koristiti za jednostavniju interakciju sa sistemom. Kako bi se takvo grafičko okruženje pokrenulo, potrebno je iskoristiti sledeću komandu:

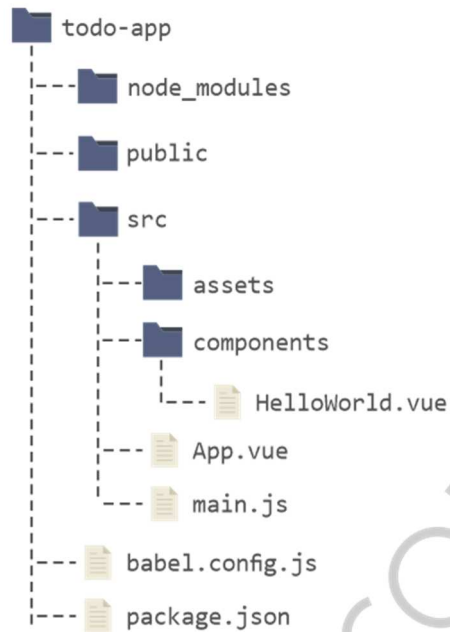
```
vue ui
```

Upućivanje ovakve komande rezultovaće otvaranjem podrazumevanog web pregledača sa stranicom koja predstavlja grafički interfejs za rukovanje Vue CLI-jem. Korišćenjem takvog okruženja moguće je obaviti sve što i direktnim upućivanjem komandi korišćenjem konzole ili terminala. Korišćenje Vue CLI grafičkog korisničkog interfejsa ilustrovano je u video-lekciji.

### Struktura Vue projekta

Vue CLI za nas automatski uspostavlja osnovnu strukturu projekta. To možete da vidite uvidom u folder unutar koga ste bili pozicionirani prilikom upućivanja komande za kreiranje novog projekta (slika 7.7).





Slika 7.7. Osnovna struktura Vue CLI projekta

Značenje kreiranih fajlova i foldera je sledeće:

- `node_modules` – folder u koji se postavljaju npm moduli koji koristi Vue CLI projekat,
- `public` – folder unutar koga se smeštaju statički fajlovi koji neće biti modifikovani, odnosno koji će da budu isporučeni u svom izvornom obliku; inicijalno, unutar ovoga foldera se nalazi `index.html` fajl, unutar čijeg tela će automatski, prilikom izgradnje da bude umetnut izgrađeni HTML kod,
- `src` – folder unutar koga se smeštaju izvorni fajlovi projekta; to su fajlovi koji se koriste prilikom razvoja, a uz pomoć kojih se obavlja izgradnja fajlova koji su spremni za to da ih koriste web pregledači,
  - `assets` – folder za smeštanje resursa, kao što su slike, vektori, ikonice i slično,
  - `components` – folder za smeštanje komponenata Vuea,
    - `HelloWorld.vue` – komponenta koja se automatski kreira prilikom kreiranja projekta,
  - `App.vue` – glavna komponenta aplikacije,
  - `main.js` – osnovni JavaScript fajl sa kodom koji obavlja povezivanje kompletnog projekta,
- `package.json` – fajl za opisivanje Vue projekta (npm manifest fajl),
- `babel.config.js` – konfiguracioni fajl Babel biblioteke.

Na osnovu upravo prikazane projektne strukture koja nastaje korišćenjem Vue CLI-ja, možemo da zaključimo nekoliko stvari.

- Vue CLI projekat je zapravo npm projekat (poseduje folder `node_modules` i `package.json` fajl koji opisuje projekat); tako dolazimo i do prvog primera realne upotrebe Node.js-a i npm-a za frontend razvoj.
- Korišćenjem Vue CLI-ja prvi put uplovljavamo u vode naprednog frontend razvoja, koji podrazumeva upotrebu specifičnih alata za izgradnju finalnog koda; tako se sada zapravo prvi put fajlovi sa izvornim kodom razlikuju od onoga što će na kraju web pregledač da dobije.
- Vue CLI projekat nije moguće direktno isporučiti kao gotov proizvod koji može da ode u produkciju; da bi takvo nešto bilo moguće, neophodno je prethodno obaviti izgradnju projekta, u čemu učestvuju različiti već spomenuti elementi Vue sistema koji se izvršavaju posredstvom izvršnog okruženja Node.js.
- Projekat kreiran korišćenjem Vue CLI-ja podrazumevano je strukturiran korišćenjem komponentata; korena komponenta aplikacije je smeštena unutar fajla `App.vue`, dok se ostale komponente, koje ćemo mi kreirati tokom razvoja aplikacije, smeštaju unutar foldera `src/components`.

## Pokretanje Vue CLI projekta

Nešto ranije je rečeno da je sastavni deo Vue CLI sistema i servis koji omogućava da se projekat pokrene i bez prethodne izgradnje, što je veoma korisno kako bi se obavilo njegovo testiranje. Projekat se pokreće posredstvom servera Node.js, koji omogućava da se projektu pristupi korišćenjem `localhost` domena. Kako bi se takav posao obavio, dovoljno je uputiti sledeću komandu:

```
npm run serve
```

Prikazana komanda se mora pokrenuti iz korenog foldera Vue projekta. Njom se npm menadžeru paketa govori da je potrebno da pokrene skriptu koja sa naziva `serve` (`npm run` komanda omogućava izvršavanje skripti). Nešto ranije je spomenut CLI Service kao sastavni deo Vue CLI sistema. Upravo je to komponenta koja omogućava izvršavanje `serve` komande (a pored nje, i izvršavanje još dve komande, o kojima će biti reči u nastavku).

Nakon izvršavanja prikazane komande, projekat će postati javno dostupan korišćenjem specijalno kreiranog lokalnog servera, na adresi:

<http://localhost:8080/>

Kada ovu adresu posetite korišćenjem web pregledača, dočekaće vas web stranica kao na slici 7.8.



# Welcome to Your Vue.js App

For a guide and recipes on how to configure / customize this project,  
check out the [vue-cli documentation](#).

## Installed CLI Plugins

[babel](#) [eslint](#)

## Essential Links

[Core Docs](#) [Forum](#) [Community Chat](#) [Twitter](#) [News](#)

## Ecosystem

[vue-router](#) [vuex](#) [vue-devtools](#) [vue-loader](#) [awesome-vue](#)

*Slika 7.8. Izgled automatski kreirane stranice našeg prvog Vue projekta kreiranog korišćenjem Vue CLI-ja*

Stranica koju vidite na slici 7.8. produkt je koda koji je nešto ranije Vue CLI automatski kreirao prilikom kreiranja projekta. Na primer, većinu koda kojim se prikazuje sadržaj na stranici možete pronaći unutar fajla `HelloWorld.vue`. Ukoliko pogledate sadržaj takvog fajla, koji predstavlja jednu Single-file Vue komponentu, možete videti sve one delove o kojima smo govorili u prethodnoj lekciji (`template`, `script`, `style`).

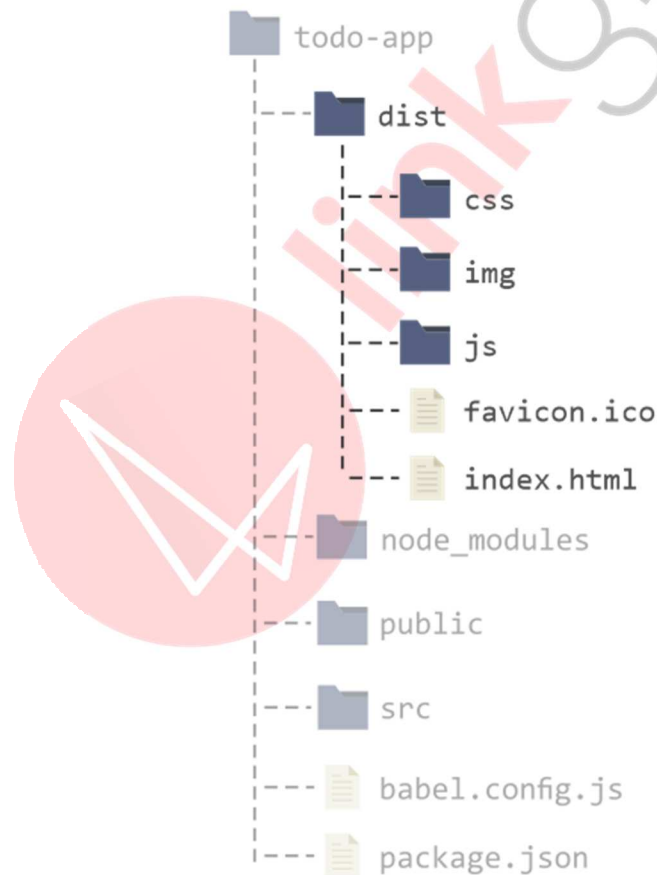
Bitno je da znate da je server koji omogućava pokretanje Vue projekta takozvani *live server*. To praktično znači da on omogućava automatsko propagiranje promena do kojih dođe na izvornom kodu. Pokušajte na primer da promenite tekst naslovne poruke unutar fajla `App.vue`. Sačuvajte izmene i onda ćete moći da vidite da je takva promena odmah vidljiva i unutar web pregledača.

## Izgradnja i objavljivanje Vue CLI projekta

Prethodno poglavlje je pokazalo kako izgleda kada Vue projekat pokreće razvojni server. Na taj način je moguće veoma brzo i lako testirati Vue aplikaciju koja se kreira. Ipak, kada se razvoj završi i kada poželimo da objavimo projekat, neophodno je pokrenuti proces izgradnje projekta. To se opet postiže korišćenjem jedne komande koju na raspolaganje nama stavlja CLI Service. Reč je o komandi `build`:

```
npm run build
```

Pokretanjem ovakve komande obaviće se proces izgradnje projekta, što podrazumeva kreiranje foldera **dist** unutar projektne strukture. Naziv `dist` je skraćenica od *distribution*, što je naziv za kod koji je spreman za realno, produkciono korišćenje (slika 7.9).



Slika 7.9. Projektna struktura nakon izgradnje projekta

Sadržaj foldera `dist` zapravo je HTML, CSS i JavaScript kod koji web pregledači mogu direktno da razumeju. Stoga sadržaj foldera `dist` direktno možete postaviti na produkcionu ili lokalni server i unutar web pregledača otvoriti Vue aplikaciju, navigacijom na `index.html` dokument.

Ipak, imajte na umu da Vue CLI podrazumevano smatra da će kreirana Vue aplikacija biti postavljena na korenu putanju nekog domena. Ukoliko želite da hostujete kreiranu Vue aplikaciju unutar nekog potfoldera, neophodno je da to naglasite unutar konfiguracionog fajla `vue.config.js`. Takav fajl je potrebno kreirati unutar korenog foldera projekta, a zatim unutar njega postaviti sledeći sadržaj:

```
module.exports = {
  publicPath: process.env.NODE_ENV === 'production'
    ? '/first-vue-app/'
    : '/'
}
```

Prikazanim kodom obavlja se definisanje vrednosti `publicPath` promenljive. Ovo svojstvo podrazumevano ima vrednost  `'/'` i upravo je to razlog zbog koga kreirana Vue aplikacija očekuje da bude smeštena unutar korenog foldera nekog domena. Problem često može nastati ukoliko želite da dobijene fajlove Vue aplikacije postavite unutar nekog potfoldera lokalnog servera (WAMP ili XAMPP), što je najčešći slučaj dok se još upoznajete sa Vue sistemom. Tada, po podrazumevanim postavkama, resursi aplikacije neće biti ispravno linkovani, pa ni aplikacija neće funkcionisati.

Promenljiva `publicPath` koristi se i prilikom hostovanja izgrađenog produkcionog projekta, ali i prilikom testiranja, hostovanjem od lokalnog Node.js servera. Upravo zbog toga je u primeru iskorišćen ternarni operator kojim se proverava režim u kome se pokreće Vue aplikacija. Ukoliko je reč o produkcionom režimu, vrednost promenljive `publicPath` se postavlja na `/first-vue-app/`, što će biti folder unutar koga će projekat biti smešten na lokalnom Apache serveru. Kada se Vue aplikacija pokrene korišćenjem komande `npm run serve`, biće korišćena korena putanja domena.

### ToDo aplikacija

Sve ono što smo naučili u lekcijama o Vue sistemu iskorišćeno je za kreiranje ToDo aplikacije. Kreiranje takve aplikacije upotrebom Vue softverskog okvira ilustrovano je u video-lekciji koja se nalazi u prilogu. Stoga ne propustite da pogledate tutorijal unutar koga ćete na praktičnom primeru moći da vidite na koji način se koristi sve ono o čemu je bilo reči u lekcijama za nama.

## Rezime

- Vue CLI je alat koji je Vue tim razvio kako bi olakšao razvoj Vue aplikacija.
- Vue CLI se instalira kao globalni npm paket, korišćenjem komande `npm install -g @vue/cli`.
- Kreiranje novog projekta korišćenjem Vue CLI-ja postiže se komandom `vue create app_name`, gde se `app_name` odnosi na naziv aplikacije koja se kreira.

- Projekti kreirani korišćenjem Vue CLI-ja podrazumevano koriste Babel i ESLint biblioteke; Babel i ESLint se koriste samo tokom razvoja, odnosno prilikom izgradnje projekta.
- Babel je JavaScript biblioteka koja je namenjena konvertovanju JavaScript koda koji je napisan uz poštovanje novijih verzija ECMAScript specifikacije u kod koji mogu izvršiti i web pregledači i izvršna okruženja koja takve nove funkcionalnosti ne podržavaju.
- ESLint je biblioteka koja se bavi analiziranjem JavaScript koda zarad pronalaska grešaka ili problematičnih delova koji ne zadovoljavaju unapred definisani standard.
- Vue CLI poseduje grafičko korisničko okruženje koje je moguće pokrenuti komandom `vue ui`.
- Vue CLI projekat nije moguće direktno isporučiti kao gotov proizvod koji može da ode u produkciju; da bi takvo nešto bilo moguće, neophodno je prethodno obaviti izgradnju projekta.
- Pokretanje Vue projekta bez izgradnje, na lokalnom Node.js serveru, postiže se komandom `npm run serve`.
- Proces izgradnje projekta, korišćenjem Vue CLI-ja, inicira se upućivanjem komande `npm run build`.
- Izgradnja Vue projekta, uz korišćenje Vue CLI-ja, podrazumeva kreiranje foldera `dist` unutar koga se smeštaju HTML, CSS i JS fajlovi, spremni za korišćenje.

