***ANGULAR***

In un progetto Angular nel file *package.json* nella sezione delle dipendenze si possono vedere le versioni utilizzate.

Le rotte sono gli indirizzi della pagina che cambiando mostrano contenuti diversi (Angular fa creare app SINGLE PAGE!).

Comandi da linea di comando:

|  |  |
| --- | --- |
| npm install -g @angular/cli | Per installare angular CLI (Command Line Interface) per creare facilmente progetti Angular. NPM è un gestore di pacchetti per javascript |
| ng version | Fa vedere la versione |
| ng new <NomeApplicazione> | Crea nuova applicazione Angular. Di base crea una pagina con indicazioni ed informazioni utili (*app.component.html* è il componente principale) |
| ng serve | Lancia il web server di default di Angular (da chiamare nella cartella di un progetto Angular. Una volta partita potete controllare su localhost:4200. Se si salva il codice si vede l'app aggiornata sul browser). |
| ng generate [component | interface |pipe | module | service] nomeElemento | Crea un elemento in automatico in una cartella dedicata (Se pipe aggiunge ‘Pipe’ alla fine del nome del file creato. Nel caso di un module o service se si mette alla fine --flat crea un singolo file e non l’intera cartella e il nome ha incluso “Module” alla fine) |
| ng add @fortawesome/angular-fontawesome@<version> | Installa il componente per usare Font Awesome. Per la versione e info varie guardare: <https://www.npmjs.com/package/@fortawesome/angular-fontawesome> |

I file *.spec.ts* sono file per i test.

Nella console del browser posso scrivere codice javascript (ad esempio creare un array di oggetti e ciclarlo mostrandolo a schermo) con: *for(let item of array) console.log(item)*

***LARAGON***

È un ambiente di sviluppo. Una volta partito se si va su *localhost* (si apre se si clicca su “Web”) c’è un link “info” che porta ad una pagina di informazioni di come è configurato. Su “Database” viene aperto Heidisql. In “Root” si accede alla cartella che contiene i file a cui si può accedere dal server. È dotato di un web server php interno. Su “Terminale” si apre un terminale coi comandi di Linux.

In “Menu/creazione veloce/laravel” è possibile creare un progetto Laravel (verrà messo nella cartella di “Root”). Oppure lo si può creare da terminale col comando “*composer create-project laravel/laravel nomeProgetto*”.

Una volta creato da terminale si può entrare nella cartella e con il comando “*php artisan serve*” si fa partire il progetto sul server di php all’indirizzo indicato (sennò si può usare il server Apache su localhost:8000. Avviare prima tutto su Laragon!!!).

In “Menu/Apache/sites-enabled” si può trovare l’opzione relativa al proprio progetto che se cliccata mostra alcune impostazioni di esso e nell’attributo S*erverName* si può vedere il nome alternativo per accedere al programma dal browser e con l’url serverName.test si accede al progetto.

Una volta creato un progetto nel terminale col comando “*mysql -u nomeUtenteMySql -p”* (-p solo se si ha una password) chiederà la password per connettersi a mysql e da là si potrà creare il db.

Nel file .env del progetto bisogna inserire i dati del db e dell’user di MySql.

Per l’autenticazione in database/seeders/DatabaseSeeder.php si è decommentato il codice in run() per poter popolare il db con 10 utenti a caso (col comando *php artisan db:seed* si fa partire il comando per popolare il db con 10 utenti). In config/app.php alla voce *faker\_locale* si sceglie la lingua degli utenti fittizi creati.

In config/cors.php si trovano le configurazioni per le chiamate a server differenti (se si chiama sé stessi va bene, sennò bisogna controllare i permessi nel cors).

Comandi terminale laragon:

|  |  |
| --- | --- |
| composer create-project laravel/laravel nomeProgetto | Crea un Progetto laravel |
| php artisan serve | Fa partire il server php integrato per il progetto nella cartella corrente all’indirizzo che viene indicato nella console |
| mysql -u nomeUtenteMySql -p | Fa accedere alla shell di MySql (-p solo se per l’utente in questione bisogna immettere la password) |
| php artisan migrate | Crea la tabella dal php |
| php artisan migrate:fresh | Serve per ricreare da zero il db eliminando le tabelle precedenti |
| php artisan db:seed | Serve per popolare la tabella col DatabaseSeeder.php |
| php artisan make:controller -r -m nomeClasseModel | Crea un controller. -r indica che è un resource controller con metodi CRUD e -m indica la classe del model alla quale è collegato. |
| php artisan route:list | Mostra le rotte presenti nell’applicazione |
| php artisan vendor:publish | Fa vedere cosa si può creare. (Con 13 fa creare le view personalizzata per gli errori). |
| php artisan | Mostra una lista di tutti i comandi possibili |

***TYPESCRIPT***

Comandi da linea di comando:

|  |  |
| --- | --- |
| tsc <NomeFile> | Compila il file *.ts* |
| tsc --watch <NomeFile> | Imposta il watch (compila automaticamente il file *.ts* quando viene modificato, per uscire ctrl-C) |
| tsc --init | Crea un file *tsconfig.json* utile quando si devono mettere molte configurazioni per la traspilazione dei file *.ts* (autocompila) |
| node <NomeFile.js> | Esegue il file javascript |