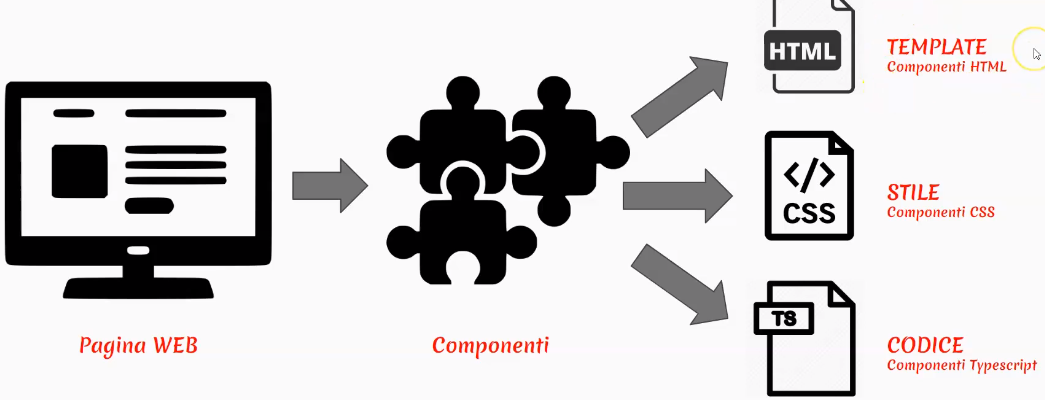
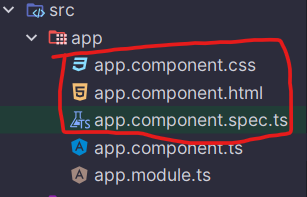
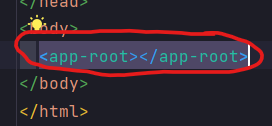
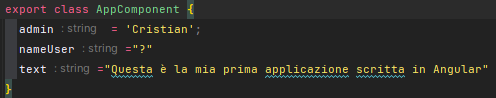
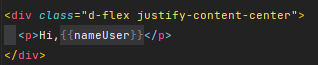
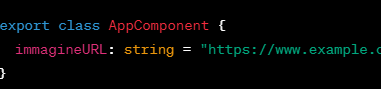
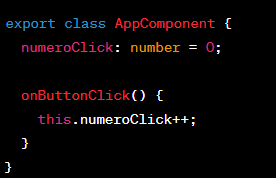
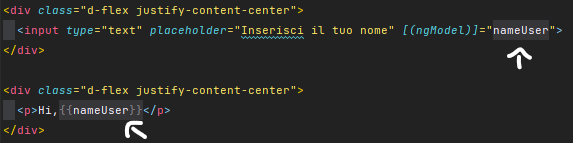
Componenti  
  
Le applicazioni in angular sono costruite da **componenti**  
  
 I componenti → quando noi visualizziamo una pagina web, questa è composta da diversi elementi come un menu, la navbar ecc, **questi vari elementi non sono altro che componenti e angular ci permette di creare questi singoli elementi separatamente.**  
  
   
**I componenti** a loro volta sono costituiti da:  
1) i template ossia la pagina html, dove è contenuto il codice html  
2) abbiamo i fogli di stile   
3) ed infine troviamo file scritti in ts dove è contenuto il codice typescript il quale viene compilato dal compilatore typescript in codice javascript il quale può essere eseguito sul browser  
**E quindi grazie all’interazione che c’è tra questi moduli creeremo la nostra pagina web**  
Perchè devo creare i componenti? → Creando i componenti divideremo il lavoro in piccolo elementi e i componenti possono essere riutilizzati nella nostra stessa applicazione o anche in altre applicazioni. **Riutilizzo del codice**

**package src.app** → Questo package contiente quei 3 file importanti tra cui  
  
   
  
**componente app.component** → Possiamo notare questi file component i quali li andiamo ad utilizzare **nel file index.html** e li utilizziamo tramite questo codice   
  
 

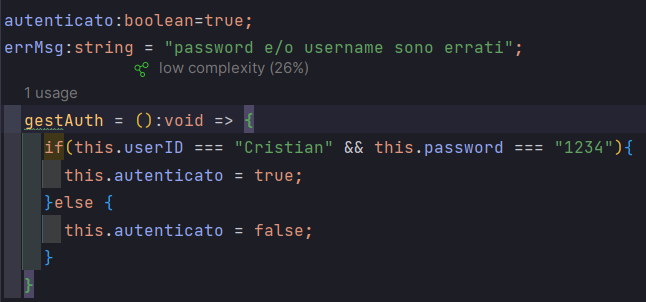
Come si crea un componente? → possiamo creare un componente da riga di comando tramite la angular cli digitando questa istruzione, la quale aggiungerà come package in src.app il componente **firstComponent,** dentro il package troveremo 3 file (html - css e ts)   
  
Nel file ts possiamo utilizzare il selettore il quale ci permette di avere accesso al componente stesso, infatti lo abbiamo utilizzato nel file index.html  
   
   
  
Scritto nel terminale questa istruzione ci creerà un nuovo componente, **skip-tests** ci permette di skippare la parte dei test, **dryrun** si accerta che non vi siano errori durante la generazione infatti, questa è una prova, **non verrà creato nulla,** successivamente abbiamo bisogno di ripetere questo comando ma senza il –dry-run  
  
 Data binding-Direttive-Moduli   
  
Esistono tre **concetti fondamentali** in angular, tra cui

**1) Data binding** → è un modo, in cui si producono i dati nel DOM, i quali dati reagiscono a eventi dell’utente. Inoltre possiamo dire che è un meccansimo che permette di sincronizzare automaticamente **i dati tra il modello** (ossia la parte logica dell’applicazione), **e la vista**. Questo permette di mantenere sempre aggiornati i dati e riflettere eventuali modifiche fatte dal modello direttamente nell’interfacia e viceversa.   
  
E un’altra caratteristica di questo meccanismo è che ci permette di salvare all’interno di una variabile, un valore proveniente dall’inserimento di qualcosa da parte dell’utente de è il caso del **binding-bidirezionale,** che però senza aver importato il modulo questo non funzionerà infatti i moduli sono uno dei tre principali concetti in angular  
 {

* intrapolation/interpolazione → {{ proprietà }} per mostrare il valore di una proprietà {{ }} **è di sola lettura** ovvero che ci permette di leggere i valori inseriti. L’interpolazione crea un collegamento unidirezionale, **quindi riusciamo a vedere il valore di quella variabile ma non possiamo modificarla.**  
     
   
* Poperty Binding → permette di assegnare il valore di una variabile a un attributo in un elemento HTML.   
     
  In questo caso l’attributo src dell’elemento img verrà impostato con il varole della variabili immagineUrl
* Event binding → consente di rispondere ad eventi dell’utente, quindi come click, tastiera collegando un evento nell’HTML a un metodo del componente  
  
* Two-way Binding o binding-bidirezionale → Combina l’event e il property binding **per mantenere sinconrizzati i dati tra il modello e la vista.**Viene implementato utilizzado la direttiva **ngModel** che consente di collegare una variabile nel modello  
    
    
  ngModel → **lettura e scrittura** in qaunto possiamo salvare dentro una variabile il valore inserito da un utente. Questa **direttiva** crea un data binding bidirezionale che ci permette di scrivere il valore della variabile in tempo reale.
* ossia nella parte logica dell’app, a un elemento HTML   
  

}

**2) Direttive** →Sono delle annotazioni nel template HTML che estendono il comportamento degli elementi HTML. Le direttive sono la via principale attraverso cui si interagisce con il DOM e si aggiunge la logica al template. Ci sono direttive incorporate in Angular, come \*ngIf, \*ngFor, \*ngSwitch, ma è anche possibile creare direttive personalizzate per aggiungere comportamenti specifici al tuo template.  
  
Le direttive sono istruzioni che vengono applicate all’interno del template di un componente angular. Esse consentono di manipolare la struttura del DOM, aggiungendo, modificando e rimuovendo elementi HTML, o modificare il loro comportamento. → {

* **Direttive Strutturali** → queste istruzioni cambiano la struttura del DOM, le quali aggiungo o rimuovono elementi HTML. Sono riconoscivili dal ( **\*** )
* **Direttive di attributo** → Queste istruzioni modificano il comportamento o l’aspetto di un elemento già esistente nel DOM  
    
   è il caso di utilizzo tipo in fase di autenticazione  
    
     
    
   
* la direttiva \*ngIf **funziona in modo che se il valore è true allora lo visualizzeremo nella pagina web** mentre se il valore è false non lo visualizziamo.

}

In sintesi, **il data binding, le direttive** **e i moduli** sono tre pilastri fondamentali di Angular che permettono di creare applicazioni dinamiche e interattive. **Il data binding sincronizza i dati tra modello e vista, mentre le direttive consentono di manipolare il DOM e personalizzare il comportamento degli elementi HTML.**

Trattiamo un altro concetto fondamentale di Angular, questo è il routing, che ci permette di far visualizzare nella stessa pagina contenuti diversi, questo fa in modo che l’utente sembri che stia cambiando pagina anche se tecnicamente rimane su quella singola pagina.  
  
 **Moduli**  
  
**3) I moduli →**  Angular, i moduli sono uno dei concetti fondamentali dell'architettura dell'applicazione. Un modulo in Angular è una collezione di componenti, direttive, servizi e altre risorse che definiscono un contesto o una funzionalità specifica all'interno dell'applicazione;   
Infatt I moduli ci permettono di **eseguire funzionalità** come il **binding bidirezionale** in cui il valore che inserisce un utente, esempio in fase di login, quel nome utente che viene inserito dev’essere catturato da angular, lo facciamo con il binding bidirezionale, ma avremo bisogno di importare il modulo che ci permetta di eseguire tale funzionalità.   
  
Quindi tutte le volte che dobbiamo aggiungere delle funzionalità e queste funzionalità risiedono in specifici moduli, dovremo andare ad importare i relativi moduli nella proprietà import del nostro NG.

