Angular simple blog

(v7.2 update)

## **Új funkció: Kommentek**

### **Feature briefing**

A blog cikk alatt szeretnénk megjeleníteni kommenteket!   
Első lépésben csak megjeleníteni szeretnénk, de később a felhasználóknak szeretnénk lehetőséget adni, hogy ők maguk is kommenteljenek, de erről később...  
  
Az új funkció összetett, fejlesztése több részre lehet bontani:

1. Design/Vázlat megértése
2. Az adatok forrása/ tárolása → adatbázis, lokális fejlesztéses esetén!
   1. mock commetek létrehozása itt: db.json
   2. comment adatszerkeszet
   3. “comments” kulcs
3. Api endpoin-tok tesztelése (postman/böngésző...)
   1. összes komment lekésére
   2. egy poszt cikk kommentjeinek lekésése
4. Új fukció architektúrális felépítése
   1. modulok
   2. komponensek
   3. service
5. Implementálás
6. Tesztelés
7. Review

# **Start**

## **Fontos** **0. lépés:** A teljes feature-t egy új branch-en hozd létre branch-et, név: *”feature/comments-listing”.* Az új feature branch a legfrissebb master/main ágból induljon ( a master-en dolgoztál/kommitoltál eddig?)

# **Design**: [angular simple blog V7.2 Comments](https://projects.invisionapp.com/freehand/document/iinu3ddd9)

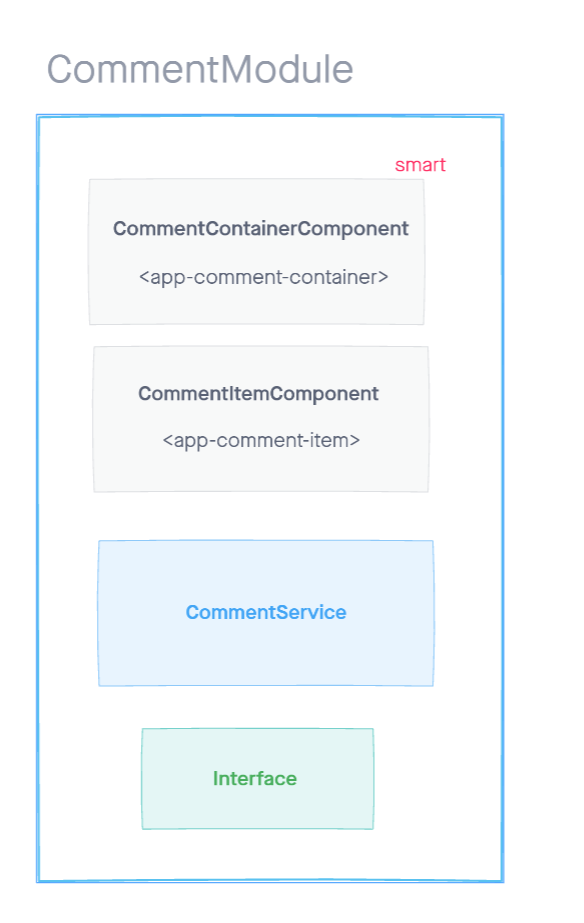
# **Adatbázis és mock adatok** Mivel jelenleg valódi szerver-oldali (alias backend) kapcsolat nélkül készítjük az alkalmazást, ezért nekünk kell gondoskodni az adatok mentéséről. A kommentek adatszerkezete az alábbi json pédán látható:

* 1. a postId.nak fontos szerepe van. Ez alapján tudjuk, hogy az adott komment melyik poszt-hoz tartozik. Pl.  
     Ha a komment postId:1, akkor a komment az 1-es id-jű poszt-hoz tartozik.
  2. A kommentnek is van saját **id**:number mezője. Az id egy egyedi azonosítóként működik, ami egyértelműen meghatározza, az egyes kommenteket! Még ha két teljesen egyforma komment is érkezik, az id mezőnek mindig kűlömbőzőnek kell lennie. Erről gondoskodik a json-server, ha POST kéréssel új komment elem kerül letárolásra, de a kezdeti elemek létrehozásánal NEked kell figyelni, hogy ne legyen két egyforma id-jű elem a json tömbben!
  3. Tesztelés végett végy fel 3 kommebtet a 2-es id.val rendelkező post-hoz!
  4. git omment “mock comments added”

1. Az adott blog cikkhez tartozó kommenteket az alábbiak szerint lehet elérni:  
     
   <local-db-server-address:port>**/comments?postId={id}**  
   Példa: Az “1”-es id-jű post cikk kommentjeihez a kérés:

<https://jsonplaceholder.typicode.com/comments?postId=1>  
  
Fontos: a postId query paraméterként megy el! ?param1=value

Ugyan az összes komment lekésére szolgáló GET endpoint-ot ( /comments) nem fogja használni az alkalmazás, teszteléshez jól jöhet.

1. Még mielőtt vadul nekiesnénk a kódolásnak vessünk egy pillantást az architekturális rajzra:  
     
     
     
     
   smart: A smart komponens konténer komponens, ő kommunikál service-ekkel, ami a biznisz adatokért felel (itt épp a kommenteket értjük biznisz adatok alatt) továbbá, megjeleníti a child/buta/dumb komponenseket, amelyek csak input és output property-ikkel kommuniálnak.   
     
   CommentContainer (smart): a modul CommentService-ét injektálva lekéri a kommenteket és kilistázza a komment elemeket.   
     
   Egy komment elem = CommentItemComponent, ami Input-ként megkap egy kommetet és megjeleníti.   
     
   **Angular elemek generálása:**
   1. Hozd létre a kommentek modulját!
      1. akár már importálhatod abba a modulba, ahol majd használod (annak a komponensenek a modujába, amelyik megjeleníti a kommntek komponensét)
   2. Hozd létre a komponenseket, de még csak létrehozzuk őket!
   3. Hozd létre a service-t, ami a kommentek lekéréseért felel majd
      1. a metódust még ne most,
      2. külön fajlba interface-t comment.model.ts
   4. B,C pontban keletkezett elemek regisztálása a modulba
      1. a modul egy komponenst exportál magából → ez kerül be a poszt cikk alá. Ez egy okos komponens.
   5. git commit “comments modul elemek készen”
2. **Angular elemek implemetálása**  
   Ha már megvan az összes szükséges építő elem, elkezdődhet az építhezés!   
   Építkezés: az adat folyás irányából, haladj azalábbi sorrendben!  
   1. Interface fájl
   2. service
      1. getComments(postId:number): Observable<Comment [ ] >
      2. git commit “inteface service kész”
   3. dumb komponent
      1. Input: item:Comment
      2. template: használj material card elemet ([mat-card](https://material.angular.io/components/card/overview))
      3. git kommit
   4. végül a smart komponent, ahol összernek a szálak!

* + 1. komponens exportáláhatóvá télele modulban, ha még nincs meg
    2. komponens elhelyzése komponens template-ben
    3. fő adat változók: pl: comments
    4. adatok lékérése post id ja alapján,
       1. meg kell kapnia a post id-t
    5. adatok megjelenítése
    6. ha nincs komment a poszthoz → design #emptyCommentsTpl
    7. git kommit “smart komp”

1. **Tesztelés**
   1. minden a helyén és málüdik a kommentek betültése
   2. id:2 post esetén megjeleníti a mock kommnteket
   3. id:1 poszt esetén → empyCommentsTpl jelenik meg
2. **Review procedúra** 
   1. minden ok → git push Githubra → uj pull request a branchből a fő ágadra (master vagy main -attól függ mire kommitoltál előtte, valeszg master)
   2. Pull request reviewrként add meg Mentoraidat! & PR linket küldd el privátba Zolinak/Laci slaack chatira!
   3. Code review by mentors
   4. commenteket adunk
   5. javítász, amit kell. Minden javítás új commit-ban a branch-re pusholva (PR frissül automatikusan...)   
      If (Minden okénak látunk)
      1. then : PR approve-t kap → branch gets merged to master → goto: Finish.
      2. else : goto 7.d

**You finished the feature!**

[**Congrats!**](http://www.blindfiveyearold.com/wp-content/uploads/2015/02/great-gatsby-whos-awesome-e1424975069140.png)

**link**

### 