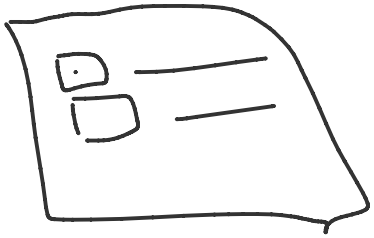
Emexo_25Nov2024

28 December 2022 10:20

1. HTML5
2. CSS3
3. JAVASCRIPT
4. TYPESCRIPT
5. ANGULAR



HTML => hypertext markup language

Hypertext markup lang => html
Hypertext

Markup lang
- Tag based (opening tag and closing tag)
- xml

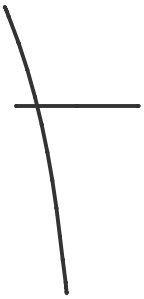
Extension => *.html

Filename . extension

Html -name of the tag
/

```
<html>
    <head></head> ---- child of of html
    <body></body> ---- child of html

</html>
```



Doctype
1. Identify the type of the document
2. To maintain consistency

```
<!DOCTYPE html>
```

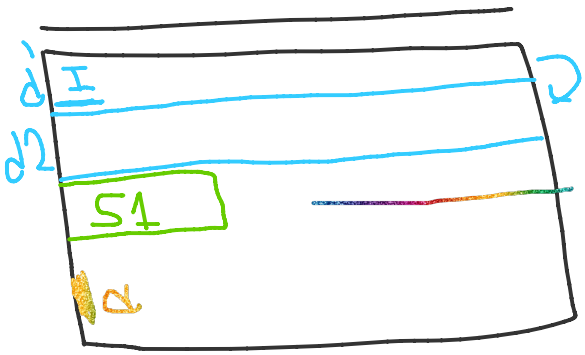
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

SEOs
title

For anchor tag href is one of the attribute

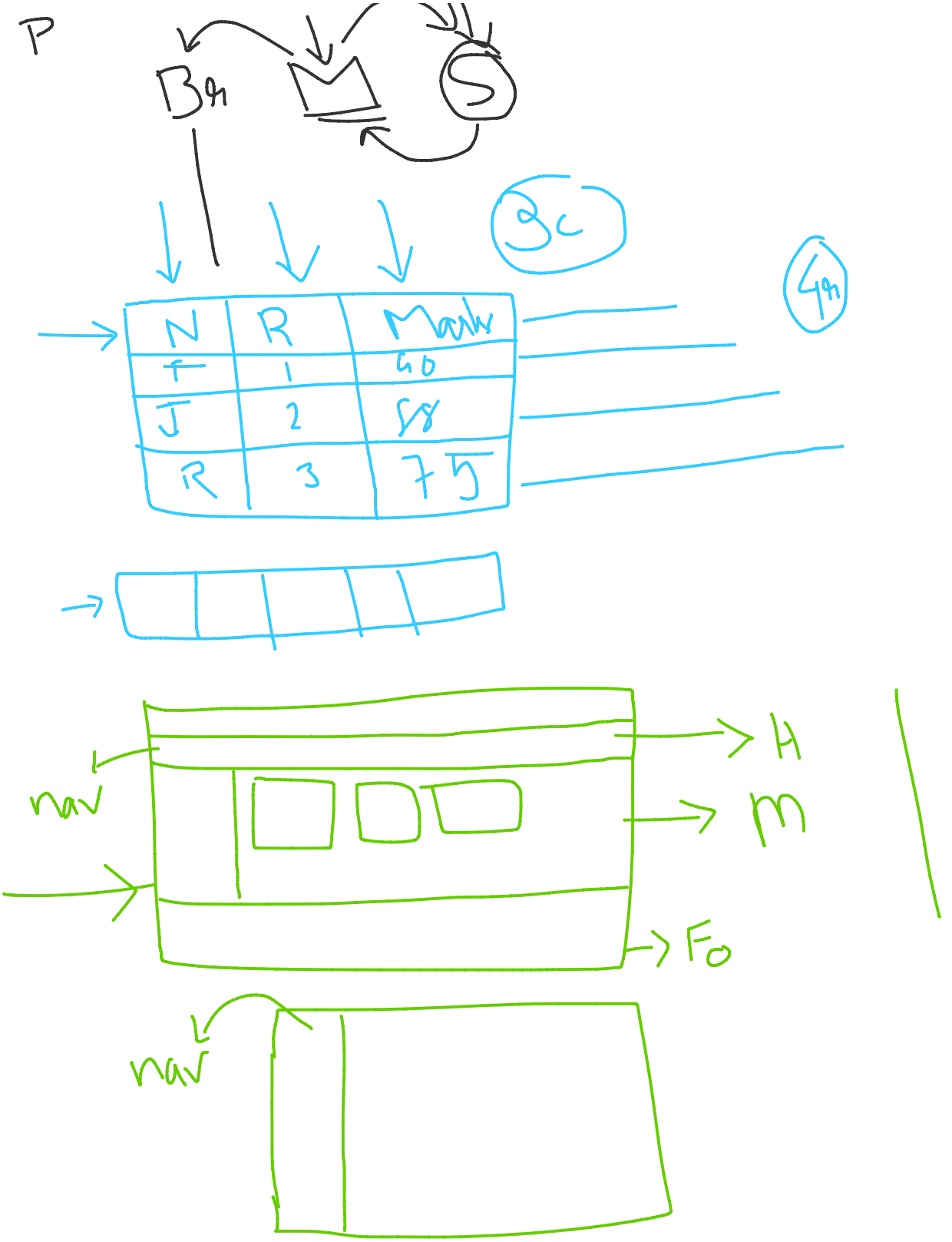
Attributes =>
It can come in most of the elements
It decides the behavior of the element
Every attribute must have a value.

Image tag
Src ===> source
Alt ===> alternate text (it comes only when image is not loaded)



Block level **elements**
1. Takes entire width of the screen

Inline elements
1. Takes width which it needs to fit in



HTML Best Practices

1. Always add document type (doc type)
2. Always use lowercase EVERY WHERE
3. Never mix upper case and lower case
4. Close all the html elements where ever applicable
5. Attribute names again should be in lower case
6. Always use quotes for the attribute values
7. When ever image is used always use alt optional to use height and width
8. Avoid space between attribute key value pair
9. Long coding lines should be avoided (200)
10. Unnecessary blank lines should be STRICTLY avoided

```
<meta http-equiv=refresh content="20">
```

```
<meta http-equiv="refresh" content="20">
```

<meta http-equiv= refresh content= 20 >

10. Unnecessary blank lines should be STRICTLY avoided
11. Never skip the title
12. Should not be omitting html/ head/ body tag
13. View port is required
14. File name should be in lower case and without any spaces
15. ***.html** or *.htm

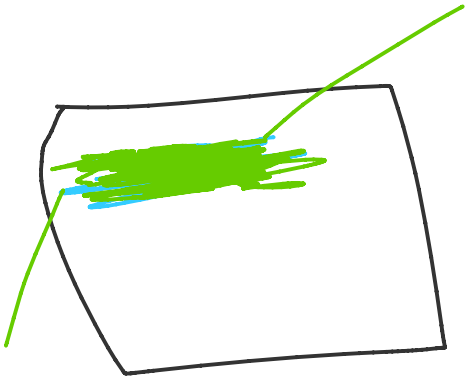
CSS3

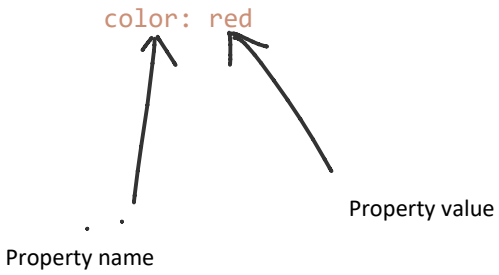
Cascading Stylesheet

*.css

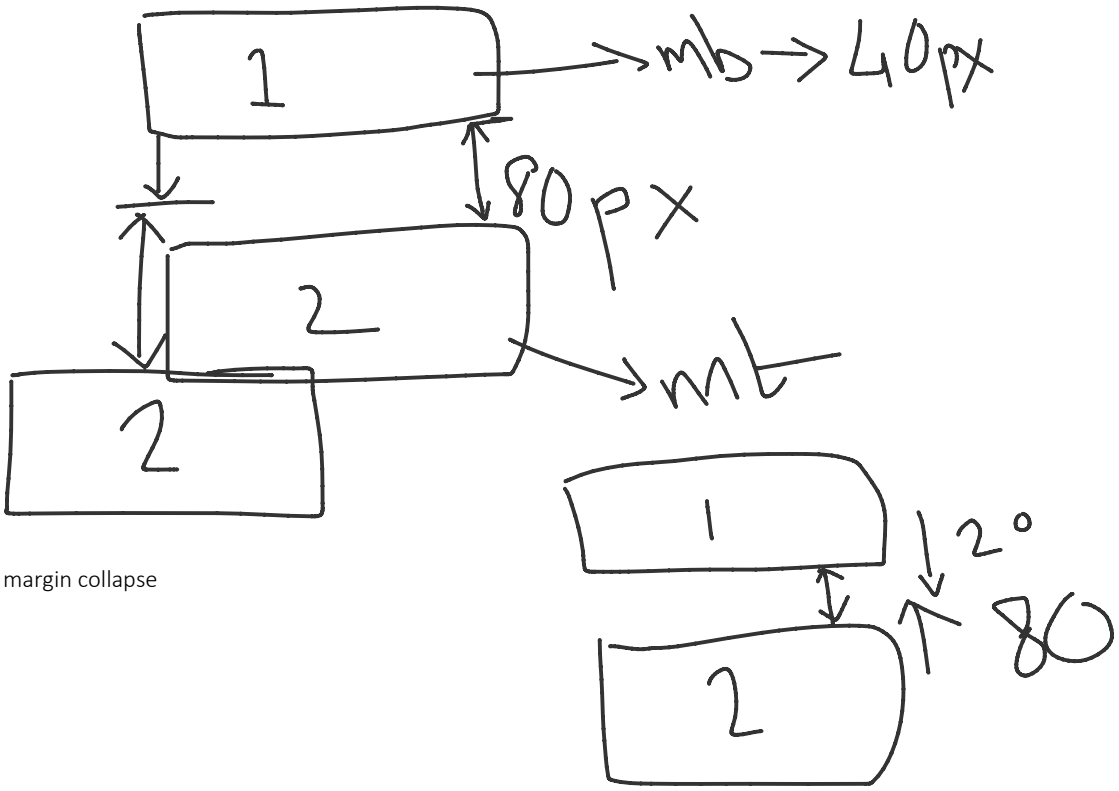
3 different ways of writing CSS

1. Inline
2. Internal
3. External

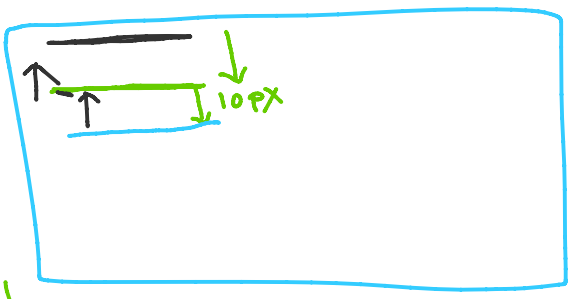


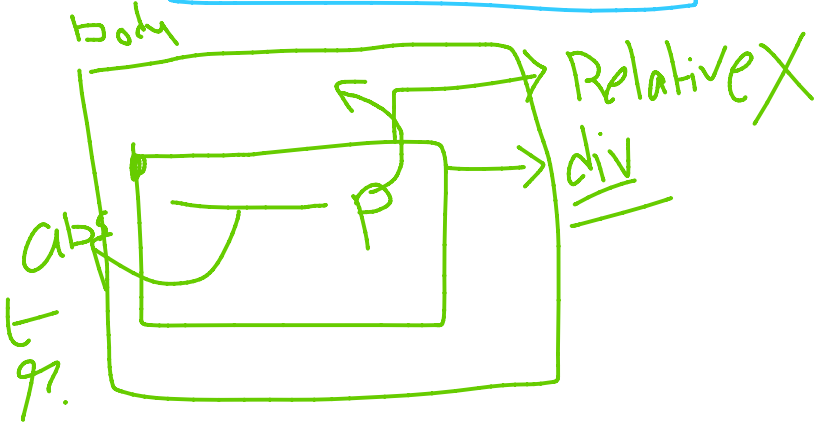


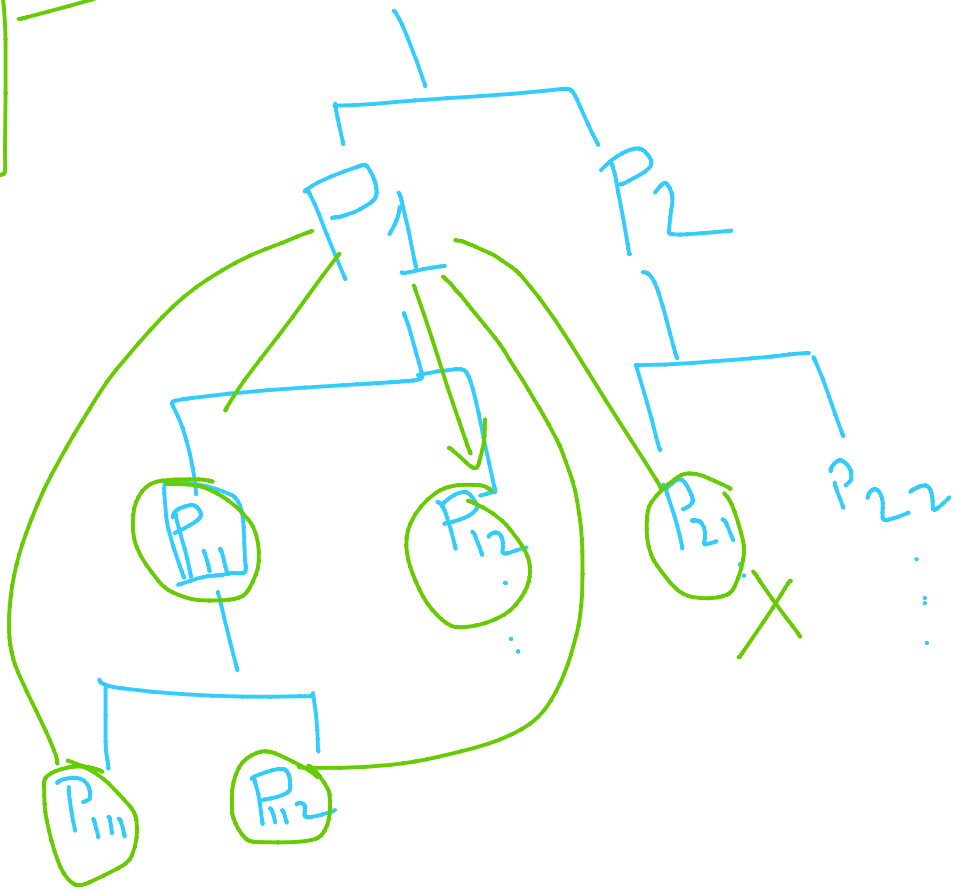
Which ever class comes last < id < inline

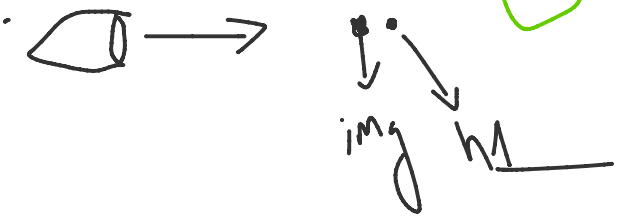


margin collapse

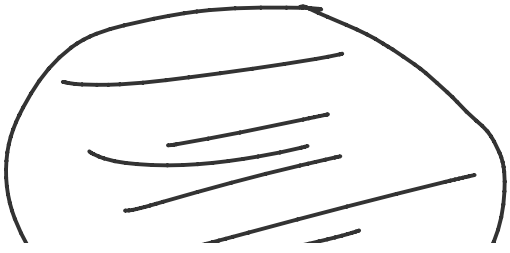


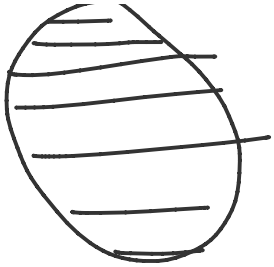






1. browser is a platform where JS runs ---- web development
2. in the server JS runs --- NODEJS creates the environment





npm - node package manager

1. Installing Angular
npm install -g @angular/cli@16

2. Verify Angular installation
ng version

3. create new project in angular
ng new <project-name>

4. Running the application
npm start

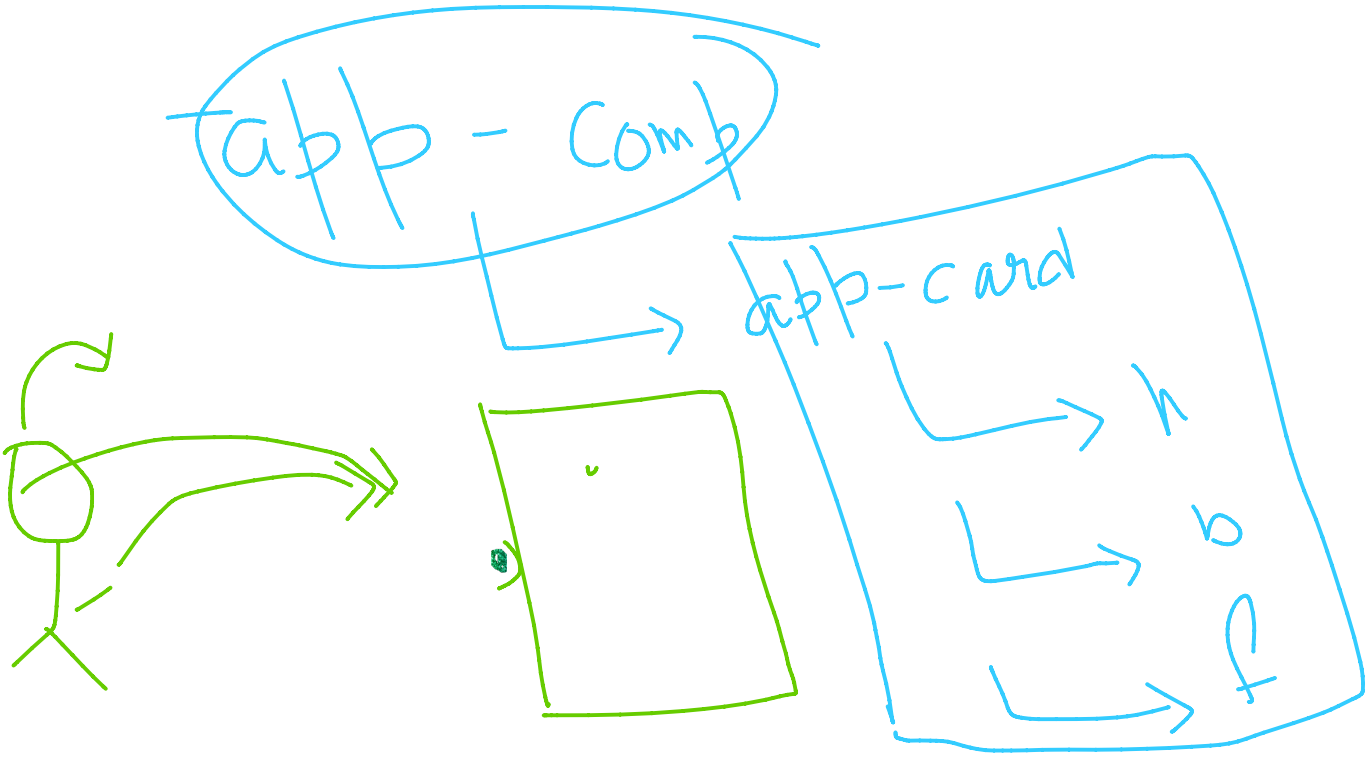
create new component
ng generate component <component-name>

ANGULAR v2 - TS framework 2015
google
routing
security
testing features
size is much more than libraries

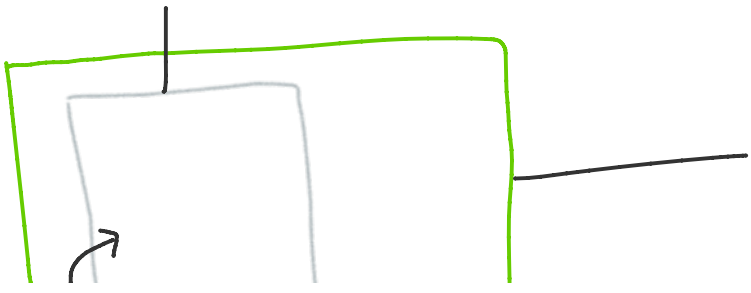
angularjs - JS - Framework (2012)
routing
spa

JQUERY - LIBRARY
smaller in size
does/ focuses on 1 single activity

1. component based architecture



main.ts ====> FIRST MODULE WHICH SHOULD BE LOADED
app.module.ts ====> FIRST COMPONENT TO BE LOADED



@Output() / EventEmitter / Child to Parent Communication

1. Define a property in the child component => p
2. Decorate with @Output()
3. The property p should be initialized with new EventEmitter()
4. There should be a method in the child component which will accept the user click
5. In this method we should use the property p to emit the event using this.p.emit()
6. (Optional) If we need to pass data then we should add/ pass the data inside the emit method. It accepts anything
7. In the parent component the event should be received using
   a. The created property should used in the component and passed a method
   b. The method should be defined inside the parent component

```
@Output()
likeClickedEvent = new EventEmitter();
```

```
Codeium: Refactor | Explain | Generate JSDoc | X
likeClicked() {
  console.log(this.title);
  console.log('Liked');
  this.likeClickedEvent.emit(this.title);
}
```

```
<app-card-footer
  [title]="title"
  (likeClickedEvent)="likeClicked($event)"
  [actions]="actions"
></app-card-footer>
/div>
```

DI

Person = XYZ

new Person()

NEW

```
DemosvcService()
```

bootstrap
tailwind
material
veneer

SPA - SINGLE PAGE APPLICATION

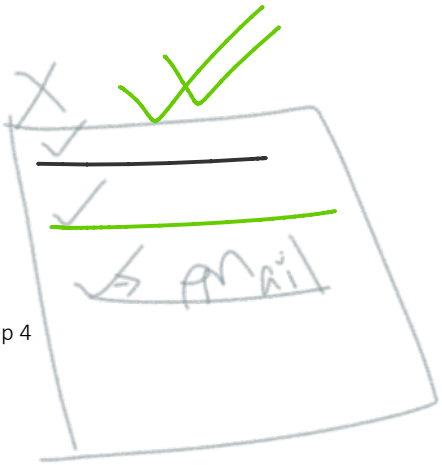
1. Need to create the components
2. in the routing module file add the routes inside the routes array
3. in the main component (app.component.html) file add the tag <router-outlet></router-outlet>

Creating a module
ng g module dashboard

http methods - {

1. POST - to create an item
2. GET - to retrieve items [] }
3. Get /id - retrieve a specific item from db
4. PUT - updating the row/ content - id
5. patch - updating - id
6. delete - deleting an item - id

6. delete - deleting an item - id



Forms

1. add the ReactiveFormsModule module in the app.module.ts file
2. In the component where the form will be added create a constructor
3. Inject the formBuilder dependency
4. create a form group/ object using the formBuilder
5. in the html add the controls with the same name in the formgroup
6. the controls should be wrapped inside a form tag
7. the form tag should have formGroup and should point to the property created in step 4