

Interview Oriented :)

# COMPLETE

# HTML

## 4 HOUR

PROJECT

CERTIFICATE

CODE



NOTES



Video Link

React.Frag

Ex- amazon Microsoft



\*  
? Levels Subscribers  
Dudig

# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Course Levels



Level 0 Setup & Fundamentals

Level 1 HTML Basics

Level 2 Must-Use HTML Tags

Level 3 Browser Tools

Level 4 HTML and Project Structure

Level 5 List, Tables & Forms

Level Bonus Github Pages & CodeSpace

# Level 0

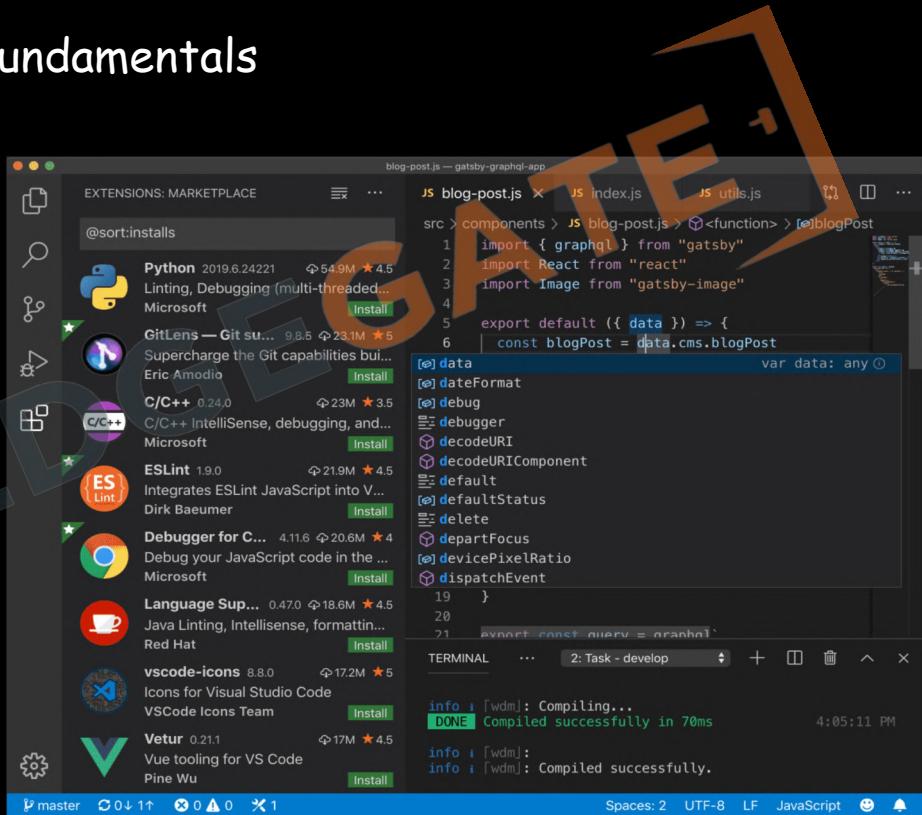
## Setup & Fundamentals

1. IDE or Code Editor
  1. What is IDE
  2. Need of IDE
  3. IDE Selection
  4. Installation and Setup
  5. VsCode Extensions
2. Website Components and Fundamentals
  1. Client Side vs Server Side
  2. FrontEnd / BackEnd / FullStack
  3. Role of Browser
  4. HTML
  5. CSS
  6. JS

# Level 0

## Setup & Fundamentals

# 1. IDE OR Code Editor



# 1.1 What is IDE

1. IDE stands for Integrated Development Environment.
2. Software suite that consolidates basic tools required for software development.
3. Central hub for coding, finding problems, and testing.
4. Designed to improve developer efficiency.



# 1.2 Need of IDE

1. Streamlines development.
2. Increases productivity.
3. Simplifies complex tasks.
4. Offers a unified workspace.
5. IDE Features
  1. Code Autocomplete
  2. Syntax Highlighting
  3. Version Control
  4. Error Checking



```
MainActivity.kt
```

```
@Composable
fun MessageCard(msg: Message) {
    Row(modifier = Modifier.padding(all = 8.dp)) {
        Image(
            painter = painterResource(R.drawable.android_studio_logo),
            contentDescription = "Profile Picture",
            modifier = Modifier
                .size(45.dp)
        )
        Spacer(modifier = Modifier.width(8.dp))
        Column (Modifier
            .background(color = Color.White)) {
            Text(text = msg.author, color = Color.Black)
            Spacer(modifier = Modifier.height(1.dp))
            Text(text = msg.body, color = Color.Black)
        }
    }
}
```



# 1.3 IDE Selection

1. Sublime Text
2. Atom
3. VS Code
4. Github CodeSpaces



The screenshot shows the VS Code Extensions Marketplace with the search bar set to "@sort:installs". The Python extension by Microsoft is listed at the top, showing it has been installed. Other extensions visible include GitLens, C/C++, ESLint, Debugger for C#, Language Support, vscode-icons, and Vetur. The main editor area shows a JavaScript file named "blog-post.js" with code related to GraphQL and Gatsby. The terminal at the bottom shows a successful compilation message.

```
blog-post.js — gatsby-graphql-app
JS blog-post.js x JS index.js JS utils.js
src > components > JS blog-post.js > <function> > blogPost
1 import { graphql } from "gatsby"
2 import React from "react"
3 import Image from "gatsby-image"
4
5 export default ({ data }) => {
6   const blogPost = data.cms.blogPost
7
8   return (
9     <div>
10       <h1>{ blogPost.title }</h1>
11       <p>{ blogPost.description }</p>
12       <img alt={ blogPost.image.alt } data-image={ blogPost.image } />
13     </div>
14   )
15 }
16
17 const query = graphql`query {
18   site {
19     siteMetadata {
20       title
21       description
22       author
23     }
24   }
25   cms {
26     blogPost {
27       id
28       title
29       description
30       image {
31         alt
32         file {
33           childImageSharp {
34             fluid {
35               ...GatsbyImageSharpFluid
36             }
37           }
38         }
39       }
40     }
41   }
42 }
43
44 export const query = graphql`query {
45   site {
46     siteMetadata {
47       title
48       description
49       author
50     }
51   }
52   cms {
53     blogPost {
54       id
55       title
56       description
57       image {
58         alt
59         file {
60           childImageSharp {
61             fluid {
62               ...GatsbyImageSharpFluid
63             }
64           }
65         }
66       }
67     }
68   }
69 }
70
71 export const query = graphql`query {
72   site {
73     siteMetadata {
74       title
75       description
76       author
77     }
78   }
79   cms {
80     blogPost {
81       id
82       title
83       description
84       image {
85         alt
86         file {
87           childImageSharp {
88             fluid {
89               ...GatsbyImageSharpFluid
90             }
91           }
92         }
93       }
94     }
95   }
96 }
97
98 const query = graphql`query {
99   site {
100     siteMetadata {
101       title
102       description
103       author
104     }
105   }
106   cms {
107     blogPost {
108       id
109       title
110       description
111       image {
112         alt
113         file {
114           childImageSharp {
115             fluid {
116               ...GatsbyImageSharpFluid
117             }
118           }
119         }
120       }
121     }
122   }
123 }
124
125 const query = graphql`query {
126   site {
127     siteMetadata {
128       title
129       description
130       author
131     }
132   }
133   cms {
134     blogPost {
135       id
136       title
137       description
138       image {
139         alt
140         file {
141           childImageSharp {
142             fluid {
143               ...GatsbyImageSharpFluid
144             }
145           }
146         }
147       }
148     }
149   }
150 }
151
152 const query = graphql`query {
153   site {
154     siteMetadata {
155       title
156       description
157       author
158     }
159   }
160   cms {
161     blogPost {
162       id
163       title
164       description
165       image {
166         alt
167         file {
168           childImageSharp {
169             fluid {
170               ...GatsbyImageSharpFluid
171             }
172           }
173         }
174       }
175     }
176   }
177 }
178
179 const query = graphql`query {
180   site {
181     siteMetadata {
182       title
183       description
184       author
185     }
186   }
187   cms {
188     blogPost {
189       id
190       title
191       description
192       image {
193         alt
194         file {
195           childImageSharp {
196             fluid {
197               ...GatsbyImageSharpFluid
198             }
199           }
200         }
201       }
202     }
203   }
204 }
205
206 const query = graphql`query {
207   site {
208     siteMetadata {
209       title
210       description
211       author
212     }
213   }
214   cms {
215     blogPost {
216       id
217       title
218       description
219       image {
220         alt
221         file {
222           childImageSharp {
223             fluid {
224               ...GatsbyImageSharpFluid
225             }
226           }
227         }
228       }
229     }
230   }
231 }
232
233 const query = graphql`query {
234   site {
235     siteMetadata {
236       title
237       description
238       author
239     }
240   }
241   cms {
242     blogPost {
243       id
244       title
245       description
246       image {
247         alt
248         file {
249           childImageSharp {
250             fluid {
251               ...GatsbyImageSharpFluid
252             }
253           }
254         }
255       }
256     }
257   }
258 }
259
260 const query = graphql`query {
261   site {
262     siteMetadata {
263       title
264       description
265       author
266     }
267   }
268   cms {
269     blogPost {
270       id
271       title
272       description
273       image {
274         alt
275         file {
276           childImageSharp {
277             fluid {
278               ...GatsbyImageSharpFluid
279             }
280           }
281         }
282       }
283     }
284   }
285 }
286
287 const query = graphql`query {
288   site {
289     siteMetadata {
290       title
291       description
292       author
293     }
294   }
295   cms {
296     blogPost {
297       id
298       title
299       description
300       image {
301         alt
302         file {
303           childImageSharp {
304             fluid {
305               ...GatsbyImageSharpFluid
306             }
307           }
308         }
309       }
310     }
311   }
312 }
313
314 const query = graphql`query {
315   site {
316     siteMetadata {
317       title
318       description
319       author
320     }
321   }
322   cms {
323     blogPost {
324       id
325       title
326       description
327       image {
328         alt
329         file {
330           childImageSharp {
331             fluid {
332               ...GatsbyImageSharpFluid
333             }
334           }
335         }
336       }
337     }
338   }
339 }
340
341 const query = graphql`query {
342   site {
343     siteMetadata {
344       title
345       description
346       author
347     }
348   }
349   cms {
350     blogPost {
351       id
352       title
353       description
354       image {
355         alt
356         file {
357           childImageSharp {
358             fluid {
359               ...GatsbyImageSharpFluid
360             }
361           }
362         }
363       }
364     }
365   }
366 }
367
368 const query = graphql`query {
369   site {
370     siteMetadata {
371       title
372       description
373       author
374     }
375   }
376   cms {
377     blogPost {
378       id
379       title
380       description
381       image {
382         alt
383         file {
384           childImageSharp {
385             fluid {
386               ...GatsbyImageSharpFluid
387             }
388           }
389         }
390       }
391     }
392   }
393 }
394
395 const query = graphql`query {
396   site {
397     siteMetadata {
398       title
399       description
400       author
401     }
402   }
403   cms {
404     blogPost {
405       id
406       title
407       description
408       image {
409         alt
410         file {
411           childImageSharp {
412             fluid {
413               ...GatsbyImageSharpFluid
414             }
415           }
416         }
417       }
418     }
419   }
420 }
421
422 const query = graphql`query {
423   site {
424     siteMetadata {
425       title
426       description
427       author
428     }
429   }
430   cms {
431     blogPost {
432       id
433       title
434       description
435       image {
436         alt
437         file {
438           childImageSharp {
439             fluid {
440               ...GatsbyImageSharpFluid
441             }
442           }
443         }
444       }
445     }
446   }
447 }
448
449 const query = graphql`query {
450   site {
451     siteMetadata {
452       title
453       description
454       author
455     }
456   }
457   cms {
458     blogPost {
459       id
460       title
461       description
462       image {
463         alt
464         file {
465           childImageSharp {
466             fluid {
467               ...GatsbyImageSharpFluid
468             }
469           }
470         }
471       }
472     }
473   }
474 }
475
476 const query = graphql`query {
477   site {
478     siteMetadata {
479       title
480       description
481       author
482     }
483   }
484   cms {
485     blogPost {
486       id
487       title
488       description
489       image {
490         alt
491         file {
492           childImageSharp {
493             fluid {
494               ...GatsbyImageSharpFluid
495             }
496           }
497         }
498       }
499     }
500   }
501 }
502
503 const query = graphql`query {
504   site {
505     siteMetadata {
506       title
507       description
508       author
509     }
510   }
511   cms {
512     blogPost {
513       id
514       title
515       description
516       image {
517         alt
518         file {
519           childImageSharp {
520             fluid {
521               ...GatsbyImageSharpFluid
522             }
523           }
524         }
525       }
526     }
527   }
528 }
529
530 const query = graphql`query {
531   site {
532     siteMetadata {
533       title
534       description
535       author
536     }
537   }
538   cms {
539     blogPost {
540       id
541       title
542       description
543       image {
544         alt
545         file {
546           childImageSharp {
547             fluid {
548               ...GatsbyImageSharpFluid
549             }
550           }
551         }
552       }
553     }
554   }
555 }
556
557 const query = graphql`query {
558   site {
559     siteMetadata {
560       title
561       description
562       author
563     }
564   }
565   cms {
566     blogPost {
567       id
568       title
569       description
570       image {
571         alt
572         file {
573           childImageSharp {
574             fluid {
575               ...GatsbyImageSharpFluid
576             }
577           }
578         }
579       }
580     }
581   }
582 }
583
584 const query = graphql`query {
585   site {
586     siteMetadata {
587       title
588       description
589       author
590     }
591   }
592   cms {
593     blogPost {
594       id
595       title
596       description
597       image {
598         alt
599         file {
600           childImageSharp {
601             fluid {
602               ...GatsbyImageSharpFluid
603             }
604           }
605         }
606       }
607     }
608   }
609 }
610
611 const query = graphql`query {
612   site {
613     siteMetadata {
614       title
615       description
616       author
617     }
618   }
619   cms {
620     blogPost {
621       id
622       title
623       description
624       image {
625         alt
626         file {
627           childImageSharp {
628             fluid {
629               ...GatsbyImageSharpFluid
630             }
631           }
632         }
633       }
634     }
635   }
636 }
637
638 const query = graphql`query {
639   site {
640     siteMetadata {
641       title
642       description
643       author
644     }
645   }
646   cms {
647     blogPost {
648       id
649       title
650       description
651       image {
652         alt
653         file {
654           childImageSharp {
655             fluid {
656               ...GatsbyImageSharpFluid
657             }
658           }
659         }
660       }
661     }
662   }
663 }
664
665 const query = graphql`query {
666   site {
667     siteMetadata {
668       title
669       description
670       author
671     }
672   }
673   cms {
674     blogPost {
675       id
676       title
677       description
678       image {
679         alt
680         file {
681           childImageSharp {
682             fluid {
683               ...GatsbyImageSharpFluid
684             }
685           }
686         }
687       }
688     }
689   }
690 }
691
692 const query = graphql`query {
693   site {
694     siteMetadata {
695       title
696       description
697       author
698     }
699   }
700   cms {
701     blogPost {
702       id
703       title
704       description
705       image {
706         alt
707         file {
708           childImageSharp {
709             fluid {
710               ...GatsbyImageSharpFluid
711             }
712           }
713         }
714       }
715     }
716   }
717 }
718
719 const query = graphql`query {
720   site {
721     siteMetadata {
722       title
723       description
724       author
725     }
726   }
727   cms {
728     blogPost {
729       id
730       title
731       description
732       image {
733         alt
734         file {
735           childImageSharp {
736             fluid {
737               ...GatsbyImageSharpFluid
738             }
739           }
740         }
741       }
742     }
743   }
744 }
745
746 const query = graphql`query {
747   site {
748     siteMetadata {
749       title
750       description
751       author
752     }
753   }
754   cms {
755     blogPost {
756       id
757       title
758       description
759       image {
760         alt
761         file {
762           childImageSharp {
763             fluid {
764               ...GatsbyImageSharpFluid
765             }
766           }
767         }
768       }
769     }
770   }
771 }
772
773 const query = graphql`query {
774   site {
775     siteMetadata {
776       title
777       description
778       author
779     }
780   }
781   cms {
782     blogPost {
783       id
784       title
785       description
786       image {
787         alt
788         file {
789           childImageSharp {
790             fluid {
791               ...GatsbyImageSharpFluid
792             }
793           }
794         }
795       }
796     }
797   }
798 }
799
800 const query = graphql`query {
801   site {
802     siteMetadata {
803       title
804       description
805       author
806     }
807   }
808   cms {
809     blogPost {
810       id
811       title
812       description
813       image {
814         alt
815         file {
816           childImageSharp {
817             fluid {
818               ...GatsbyImageSharpFluid
819             }
820           }
821         }
822       }
823     }
824   }
825 }
826
827 const query = graphql`query {
828   site {
829     siteMetadata {
830       title
831       description
832       author
833     }
834   }
835   cms {
836     blogPost {
837       id
838       title
839       description
840       image {
841         alt
842         file {
843           childImageSharp {
844             fluid {
845               ...GatsbyImageSharpFluid
846             }
847           }
848         }
849       }
850     }
851   }
852 }
853
854 const query = graphql`query {
855   site {
856     siteMetadata {
857       title
858       description
859       author
860     }
861   }
862   cms {
863     blogPost {
864       id
865       title
866       description
867       image {
868         alt
869         file {
870           childImageSharp {
871             fluid {
872               ...GatsbyImageSharpFluid
873             }
874           }
875         }
876       }
877     }
878   }
879 }
880
881 const query = graphql`query {
882   site {
883     siteMetadata {
884       title
885       description
886       author
887     }
888   }
889   cms {
890     blogPost {
891       id
892       title
893       description
894       image {
895         alt
896         file {
897           childImageSharp {
898             fluid {
899               ...GatsbyImageSharpFluid
900             }
901           }
902         }
903       }
904     }
905   }
906 }
907
908 const query = graphql`query {
909   site {
910     siteMetadata {
911       title
912       description
913       author
914     }
915   }
916   cms {
917     blogPost {
918       id
919       title
920       description
921       image {
922         alt
923         file {
924           childImageSharp {
925             fluid {
926               ...GatsbyImageSharpFluid
927             }
928           }
929         }
930       }
931     }
932   }
933 }
934
935 const query = graphql`query {
936   site {
937     siteMetadata {
938       title
939       description
940       author
941     }
942   }
943   cms {
944     blogPost {
945       id
946       title
947       description
948       image {
949         alt
950         file {
951           childImageSharp {
952             fluid {
953               ...GatsbyImageSharpFluid
954             }
955           }
956         }
957       }
958     }
959   }
960 }
961
962 const query = graphql`query {
963   site {
964     siteMetadata {
965       title
966       description
967       author
968     }
969   }
970   cms {
971     blogPost {
972       id
973       title
974       description
975       image {
976         alt
977         file {
978           childImageSharp {
979             fluid {
980               ...GatsbyImageSharpFluid
981             }
982           }
983         }
984       }
985     }
986   }
987 }
988
989 const query = graphql`query {
990   site {
991     siteMetadata {
992       title
993       description
994       author
995     }
996   }
997   cms {
998     blogPost {
999       id
1000       title
1001       description
1002       image {
1003         alt
1004         file {
1005           childImageSharp {
1006             fluid {
1007               ...GatsbyImageSharpFluid
1008             }
1009           }
1010         }
1011       }
1012     }
1013   }
1014 }
1015
1016 const query = graphql`query {
1017   site {
1018     siteMetadata {
1019       title
1020       description
1021       author
1022     }
1023   }
1024   cms {
1025     blogPost {
1026       id
1027       title
1028       description
1029       image {
1030         alt
1031         file {
1032           childImageSharp {
1033             fluid {
1034               ...GatsbyImageSharpFluid
1035             }
1036           }
1037         }
1038       }
1039     }
1040   }
1041 }
1042
1043 const query = graphql`query {
1044   site {
1045     siteMetadata {
1046       title
1047       description
1048       author
1049     }
1050   }
1051   cms {
1052     blogPost {
1053       id
1054       title
1055       description
1056       image {
1057         alt
1058         file {
1059           childImageSharp {
1060             fluid {
1061               ...GatsbyImageSharpFluid
1062             }
1063           }
1064         }
1065       }
1066     }
1067   }
1068 }
1069
1070 const query = graphql`query {
1071   site {
1072     siteMetadata {
1073       title
1074       description
1075       author
1076     }
1077   }
1078   cms {
1079     blogPost {
1080       id
1081       title
1082       description
1083       image {
1084         alt
1085         file {
1086           childImageSharp {
1087             fluid {
1088               ...GatsbyImageSharpFluid
1089             }
1090           }
1091         }
1092       }
1093     }
1094   }
1095 }
1096
1097 const query = graphql`query {
1098   site {
1099     siteMetadata {
1100       title
1101       description
1102       author
1103     }
1104   }
1105   cms {
1106     blogPost {
1107       id
1108       title
1109       description
1110       image {
1111         alt
1112         file {
1113           childImageSharp {
1114             fluid {
1115               ...GatsbyImageSharpFluid
1116             }
1117           }
1118         }
1119       }
1120     }
1121   }
1122 }
1123
1124 const query = graphql`query {
1125   site {
1126     siteMetadata {
1127       title
1128       description
1129       author
1130     }
1131   }
1132   cms {
1133     blogPost {
1134       id
1135       title
1136       description
1137       image {
1138         alt
1139         file {
1140           childImageSharp {
1141             fluid {
1142               ...GatsbyImageSharpFluid
1143             }
1144           }
1145         }
1146       }
1147     }
1148   }
1149 }
1150
1151 const query = graphql`query {
1152   site {
1153     siteMetadata {
1154       title
1155       description
1156       author
1157     }
1158   }
1159   cms {
1160     blogPost {
1161       id
1162       title
1163       description
1164       image {
1165         alt
1166         file {
1167           childImageSharp {
1168             fluid {
1169               ...GatsbyImageSharpFluid
1170             }
1171           }
1172         }
1173       }
1174     }
1175   }
1176 }
1177
1178 const query = graphql`query {
1179   site {
1180     siteMetadata {
1181       title
1182       description
1183       author
1184     }
1185   }
1186   cms {
1187     blogPost {
1188       id
1189       title
1190       description
1191       image {
1192         alt
1193         file {
1194           childImageSharp {
1195             fluid {
1196               ...GatsbyImageSharpFluid
1197             }
1198           }
1199         }
1200       }
1201     }
1202   }
1203 }
1204
1205 const query = graphql`query {
1206   site {
1207     siteMetadata {
1208       title
1209       description
1210       author
1211     }
1212   }
1213   cms {
1214     blogPost {
1215       id
1216       title
1217       description
1218       image {
1219         alt
1220         file {
1221           childImageSharp {
1222             fluid {
1223               ...GatsbyImageSharpFluid
1224             }
1225           }
1226         }
1227       }
1228     }
1229   }
1230 }
1231
1232 const query = graphql`query {
1233   site {
1234     siteMetadata {
1235       title
1236       description
1237       author
1238     }
1239   }
1240   cms {
1241     blogPost {
1242       id
1243       title
1244       description
1245       image {
1246         alt
1247         file {
1248           childImageSharp {
1249             fluid {
1250               ...GatsbyImageSharpFluid
1251             }
1252           }
1253         }
1254       }
1255     }
1256   }
1257 }
1258
1259 const query = graphql`query {
1260   site {
1261     siteMetadata {
1262       title
1263       description
1264       author
1265     }
1266   }
1267   cms {
1268     blogPost {
1269       id
1270       title
1271       description
1272       image {
1273         alt
1274         file {
1275           childImageSharp {
1276             fluid {
1277               ...GatsbyImageSharpFluid
1278             }
1279           }
1280         }
1281       }
1282     }
1283   }
1284 }
1285
1286 const query = graphql`query {
1287   site {
1288     siteMetadata {
1289       title
1290       description
1291       author
1292     }
1293   }
1294   cms {
1295     blogPost {
1296       id
1297       title
1298       description
1299       image {
1300         alt
1301         file {
1302           childImageSharp {
1303             fluid {
1304               ...GatsbyImageSharpFluid
1305             }
1306           }
1307         }
1308       }
1309     }
1310   }
1311 }
1312
1313 const query = graphql`query {
1314   site {
1315     siteMetadata {
1316       title
1317       description
1318       author
1319     }
1320   }
1321   cms {
1322     blogPost {
1323       id
1324       title
1325       description
1326       image {
1327         alt
1328         file {
1329           childImageSharp {
1330             fluid {
1331               ...GatsbyImageSharpFluid
1332             }
1333           }
1334         }
1335       }
1336     }
1337   }
1338 }
1339
1340 const query = graphql`query {
1341   site {
1342     siteMetadata {
1343       title
1344       description
1345       author
1346     }
1347   }
1348   cms {
1349     blogPost {
1350       id
1351       title
1352       description
1353       image {
1354         alt
1355         file {
1356           childImageSharp {
1357             fluid {
1358               ...GatsbyImageSharpFluid
1359             }
1360           }
1361         }
1362       }
1363     }
1364   }
1365 }
1366
1367 const query = graphql`query {
1368   site {
1369     siteMetadata {
1370       title
1371       description
1372       author
1373     }
1374   }
1375   cms {
1376     blogPost {
1377       id
1378       title
1379       description
1380       image {
1381         alt
1382         file {
1383           childImageSharp {
1384             fluid {
1385               ...GatsbyImageSharpFluid
1386             }
1387           }
1388         }
1389       }
1390     }
1391   }
1392 }
1393
1394 const query = graphql`query {
1395   site {
1396     siteMetadata {
1397       title
1398       description
1399       author
1400     }
1401   }
1402   cms {
1403     blogPost {
1404       id
1405       title
1406       description
1407       image {
1408         alt
1409         file {
1410           childImageSharp {
1411             fluid {
1412               ...GatsbyImageSharpFluid
1413             }
1414           }
1415         }
1416       }
1417     }
1418   }
1419 }
1420
1421 const query = graphql`query {
1422   site {
1423     siteMetadata {
1424       title
1425       description
1426       author
1427     }
1428   }
1429   cms {
1430     blogPost {
1431       id
1432       title
1433       description
1434       image {
1435         alt
1436         file {
1437           childImageSharp {
1438             fluid {
1439               ...GatsbyImageSharpFluid
1440             }
1441           }
1442         }
1443       }
1444     }
1445   }
1446 }
1447
1448 const query = graphql`query {
1449   site {
1450     siteMetadata {
1451       title
1452       description
1453       author
1454     }
1455   }
1456   cms {
1457     blogPost {
1458       id
1459       title
1460       description
1461       image {
1462         alt
1463         file {
1464           childImageSharp {
1465             fluid {
1466               ...GatsbyImageSharpFluid
1467             }
1468           }
1469         }
1470       }
1471     }
1472   }
1473 }
1474
1475 const query = graphql`query {
1476   site {
1477     siteMetadata {
1478       title
1479       description
1480       author
1481     }
1482   }
1483   cms {
1484     blogPost {
1485       id
1486       title
1487       description
1488       image {
1489         alt
1490         file {
1491           childImageSharp {
1492             fluid {
1493               ...GatsbyImageSharpFluid
1494             }
1495           }
1496         }
1497       }
1498     }
1499   }
1500 }
1501
1502 const query = graphql`query {
1503   site {
1504     siteMetadata {
1505       title
1506       description
1507       author
1508     }
1509   }
1510   cms {
1511     blogPost {
1512       id
1513       title
1514       description
1515       image {
1516         alt
1517         file {
1518           childImageSharp {
1519             fluid {
1520               ...GatsbyImageSharpFluid
1521             }
1522           }
1523         }
1524       }
1525     }
1526   }
1527 }
1528
1529 const query = graphql`query {
1530   site {
1531     siteMetadata {
1532       title
1533       description
1534       author
1535     }
1536   }
1537   cms {
1538     blogPost {
1539       id
1540       title
1541       description
1542       image {
1543         alt
1544         file {
1545           childImageSharp {
1546             fluid {
1547               ...GatsbyImageSharpFluid
1548             }
1549           }
1550         }
1551       }
1552     }
1553   }
1554 }
1555
1556 const query = graphql`query {
1557   site {
1558     siteMetadata {
1559       title
1560       description
1561       author
1562     }
1563   }
1564   cms {
1565     blogPost {
1566       id
1567       title
1568       description
1569       image {
1570         alt
1571         file {
1572           childImageSharp {
1573             fluid {
1574               ...GatsbyImageSharpFluid
1575             }
1576           }
1577         }
1578       }
1579     }
1580   }
1581 }
1582
1583 const query = graphql`query {
1584   site {
1585     siteMetadata {
1586       title
1587       description
1588       author
1589     }
1590   }
1591   cms {
1592     blogPost {
1593       id
1594       title
1595       description
1596       image {
1597         alt
1598         file {
1599           childImageSharp {
1600             fluid {
1601               ...GatsbyImageSharpFluid
1602             }
1603           }
1604         }
1605       }
1606     }
1607   }
1608 }
1609
1610 const query = graphql`query {
1611   site {
1612     siteMetadata {
1613       title
1614       description
1615       author
1616     }
1617   }
1618   cms {
1619     blogPost {
1620       id
1621       title
1622       description
1623       image {
1624         alt
1625         file {
1626           childImageSharp {
1627             fluid {
1628               ...GatsbyImageSharpFluid
1629             }
1630           }
1631         }
1632       }
1633     }
1634   }
1635 }
1636
1637 const query = graphql`query {
1638   site {
1639     siteMetadata {
1640       title
1641       description
1642       author
1643     }
1644   }
1645   cms {
1646     blogPost {
1647       id
1648       title
1649       description
1650       image {
1651         alt
1652         file {
1653           childImageSharp {
1654             fluid {
1655               ...GatsbyImageSharpFluid
1656             }
1657           }
1658         }
1659       }
1660     }
1661   }
1662 }
1663
1664 const query = graphql`query {
1665   site {
1666     siteMetadata {
1667       title
1668       description
1669       author
1670     }
1671   }
1672   cms {
1673     blogPost {
1674       id
1675       title
1676       description
1677       image {
1678         alt
1679         file {
1680           childImageSharp {
1681             fluid {
1682               ...GatsbyImageSharpFluid
1683             }
1684           }
1685         }
1686       }
1687     }
1688   }
1689 }
1690
1691 const query = graphql`query {
1692   site {
1693     siteMetadata {
1694       title
1695       description
1696       author
1697     }
1698   }
1699   cms {
1700     blogPost {
1701       id
1702       title
1703       description
1704       image {
1705         alt
1706         file {
1707           childImageSharp {
1708             fluid {
1709               ...GatsbyImageSharpFluid
1710             }
1711           }
1712         }
1713       }
1714     }
1715   }
1716 }
1717
1718 const query = graphql`query {
1719   site {
1720     siteMetadata {
1721       title
1722       description
1723       author
1724     }
1725   }
1726   cms {
1727     blogPost {
1728       id
1729       title
1730       description
1731       image {
1732         alt
1733         file {
1734           childImageSharp {
1735             fluid {
1736               ...GatsbyImageSharpFluid
1737             }
1738           }
1739         }
1740       }
1741     }
1742   }
1743 }
1744
1745 const query = graphql`query {
1746   site {
1747     siteMetadata {
1748       title
1749       description
1750       author
1751     }
1752   }
1753   cms {
1754     blogPost {
1755       id
1756       title
1757       description
1758       image {
1759         alt
1760         file {
1761           childImageSharp {
1762             fluid {
1763               ...GatsbyImageSharpFluid
1764             }
1765           }
1766         }
1767       }
1768     }
1769   }
1770 }
1771
1772 const query = graphql`query {
1773   site {
1774     siteMetadata {
1775       title
1776       description
1777       author
1778     }
1779   }
1780   cms {
1781     blogPost {
1782       id
1783       title
1784       description
1785       image {
1786         alt
1787         file {
1788           childImageSharp {
1789             fluid {
1790               ...GatsbyImageSharpFluid
1791             }
1792           }
1793         }
1794       }
1795     }
1796   }
1797 }
1798
1799 const query = graphql`query {
1800   site {
1801     siteMetadata {
1802       title
1803       description
1804       author
1805     }
1806   }
1807   cms {
1808     blogPost {
1809       id
1810       title
1811       description
1812       image {
1813         alt
1814         file {
1815           childImageSharp {
1816             fluid {
1817               ...GatsbyImageSharpFluid
1818             }
1819           }
1820         }
1821       }
1822     }
1823   }
1824 }
1825
1826 const query = graphql`query {
1827   site {
1828     siteMetadata {
1829       title
1830       description
1831       author
1832     }
1833   }
1834   cms {
1835     blogPost {
1836       id
1837       title
1838       description
1839       image {
1840         alt
1841         file {
1842           childImageSharp {
1843
```



# 1.4 Installation & Setup

## 1. Search VS Code

KNOWLEDGE GATE



# 1.5 VsCode Extensions

External Packages . Default VSC is Light, Extensions for Different DEV.

1. Live Server      Local Server,Llive changes
2. Prettier      Code Formatting Auto



View>Word Wrap  
File>Auto Save



# Level 0

Setup & Fundamentals

## 2. Website Components And Fundamentals



# 2.1 Client Side vs Server Side

	Client Side	Server Side
Execution Location	Executes on user's device.	Executes on a remote machine.
Languages	Primarily JavaScript, HTML, CSS.	PHP, Python, Java, Node.js, etc.
Main Job	Makes clicks and scrolls work	Manages saved information
Access Level	Can't access server data directly	Can read/write files, interact with databases.
Speed	Quicker for UI changes	Slower due to network latency.

# 2.2 FrontEnd / BackEnd / FullStack



Client Side / Front-End  
Web Development

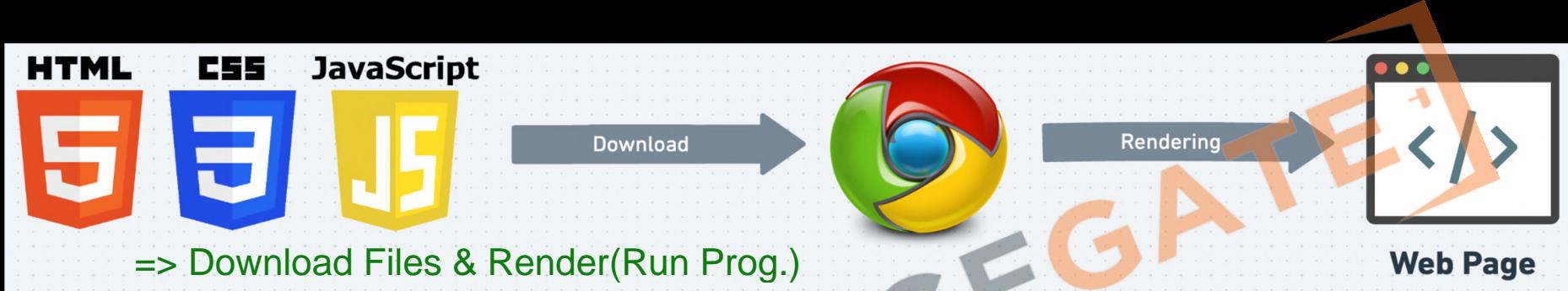


Server Side  
Back-End



Full Stack

# 2.3 Role of Browser



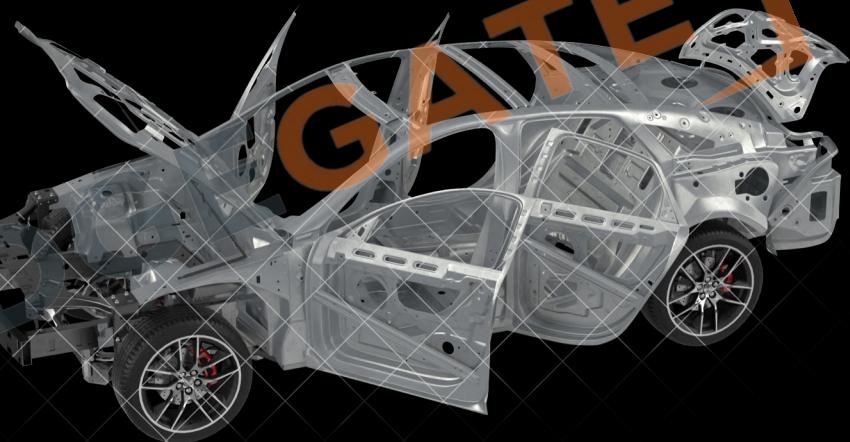
1. Displays Web Page: Turns HTML code into what you see on screen.
2. User Clicks: Helps you interact with the web page.
3. Updates Content: Allows changes to the page using JavaScript.
4. Loads Files: Gets HTML, images, etc., from the server.



# 2.4 HTML

(Hypertext Markup Language)

1. **Structure:** Sets up the layout.
2. **Content:** Adds text, images, links.
3. **Tags:** Uses elements like `<p>`, `<a>`.
4. **Hierarchy:** Organizes elements in a tree.



Yaar Mujhe Theory nhi aati,...But terese jayda accha proj. hai



# 2.5 CSS

(Cascading Style Sheets)

1. **Style:** Sets the look and feel.
2. **Colors & Fonts:** Customizes text and background.
3. **Layout:** Controls position and size.
4. **Selectors:** Targets specific **HTML** elements.





# 2.6 JS

Aatmaa  
(Java Script)

1. JavaScript has nothing to do with Java
2. Actions: Enables interactivity.
3. Updates: Alters page without reloading.
4. Events: Responds to user actions.
5. Data: Fetches and sends info to server.



# Level 0 Revision

## Setup & Fundamentals

1. IDE or Code Editor
  1. What is IDE
  2. Need of IDE
  3. IDE Selection
  4. Installation and Setup
  5. VsCode Extensions
2. Website Components and Fundamentals
  1. Client Side vs Server Side
  2. FrontEnd / BackEnd / FullStack
  3. Role of Browser
  4. HTML
  5. CSS
  6. JS



EDGATE

# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

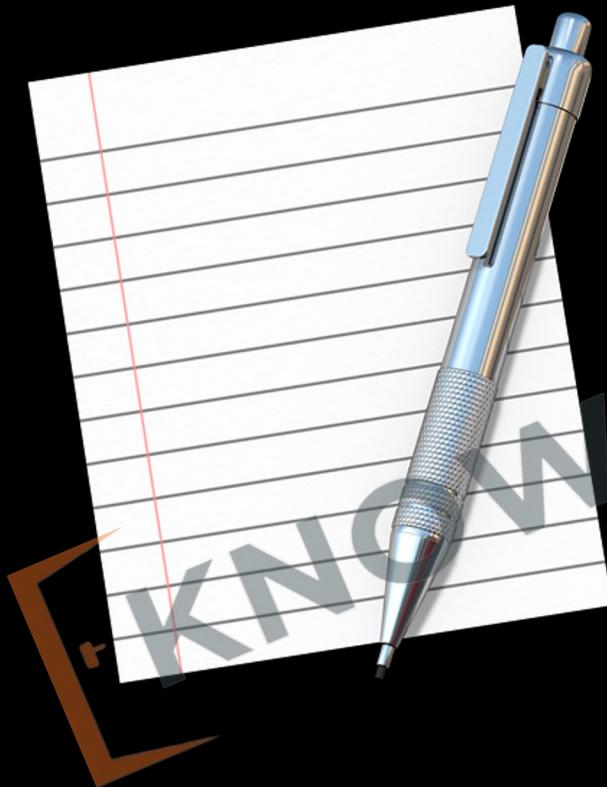
# Level 1

## HTML Basics

1. Starting up
  1. First File using Text Editor
  2. File Extension
  3. Opening the project in VsCode
  4. Index.html
2. Basics of HTML
  1. What are Tags
  2. Using Emmet ! to generate code
  3. Basic HTML Page
  4. MDN Documentation
  5. Comments
  6. Case Sensitivity

# Level 1

HTML Basics



## 1. Starting Up

KNOWLEDGE STATE

# 1.1 First file using Text Editor

1. Create a folder with name **First Project** on your Desktop.
2. Open **Notepad**.
3. Create a file and save it as **index.html**
4. Copy Sample code
5. Open **Browser** and Check.

```
<!DOCTYPE html>
<html>
<head><title></title></head>
<body>
  <p>Boilerplate</p>
</body>
</html>
```

127.0.0.1 => Local Address



# 1.2 File Extension

## HTML

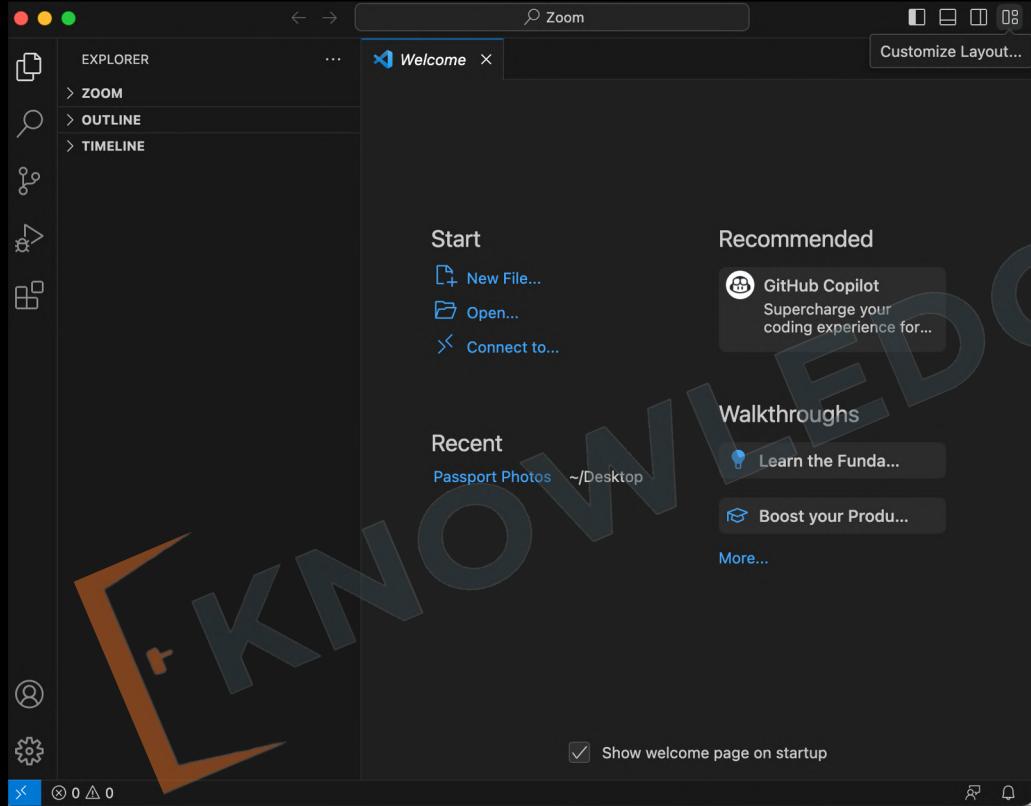
1. Most commonly used.
2. Works across all browsers.
3. Widely recognized and supported.
4. Typically saved as .html.



## HTM

1. Less commonly used.
2. Originated for compatibility with older systems.
3. Works same as .html.
4. Typically saved as .htm.

# 1.3 Opening project in VsCode



KNOWLEDGE  
DEGATE

# 1.4 Importance of index.html

1. Default name of a website's homepage.
2. First page users see when visiting a website
3. Important for SEO (Search Engine Optimization)
4. Provides uniform starting point across servers
5. Serves as fallback when no file is specified in URL



# Level 1

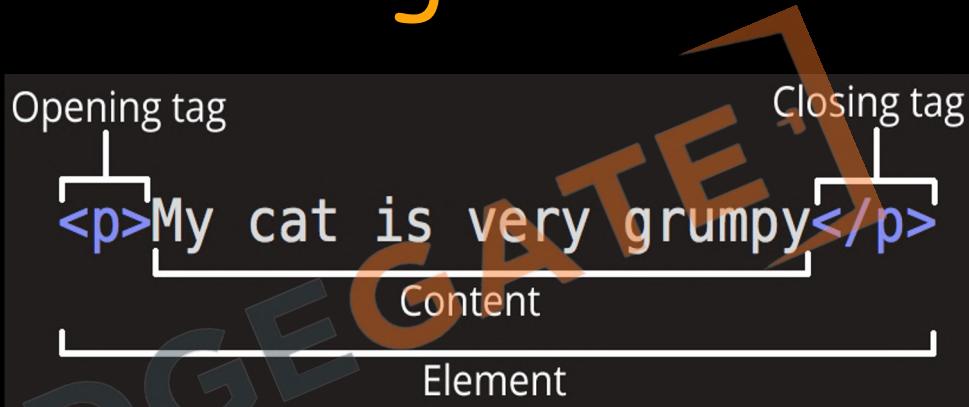
## HTML Basics

### 2. Basics of HTML



# 2.1 What are Tags

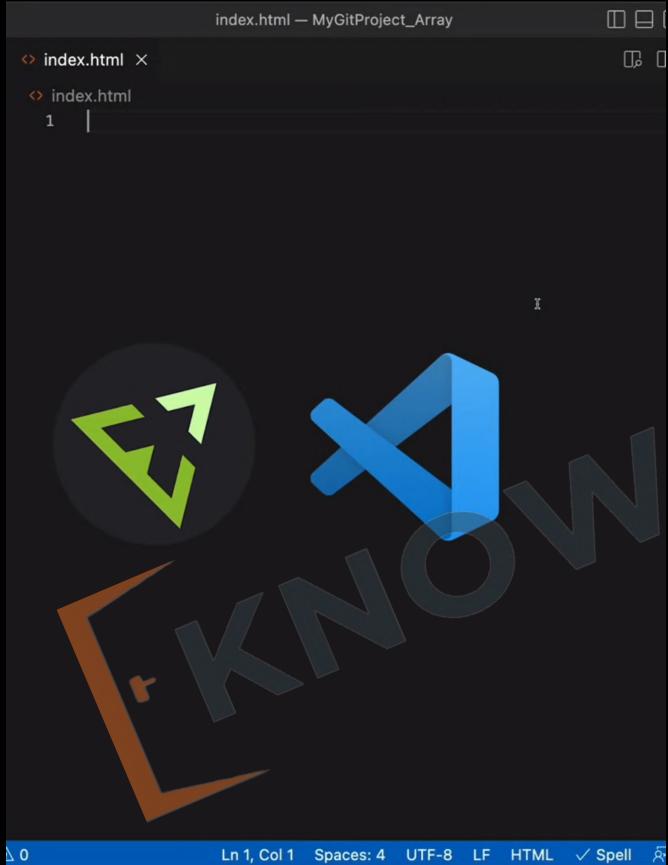
1. Elements that are used to create a website are called HTML Tags.
2. Tags can contain content or other HTML tags.
3. Define elements like text, images, links



# 2.2 Using Emmet ! to generate code

By Default Functionality by VSC

1. Type ! and wait for suggestions.



# 2.3 Basic HTML Page

```
<!DOCTYPE html>  
  
<html lang="en">  
  
    <head>  
        <title>My First Webpage</title>  
    </head>  
  
    <body>  
        <h1>Hello World!</h1>  
    </body>  
  
</html>
```

Defines the HTML Version

it tells to the browser

Parent of all HTML tags / Root element

Parent of meta data tags

Title of the web page

Meta Data: manual of Washing machine ,i.e. Booklet of data

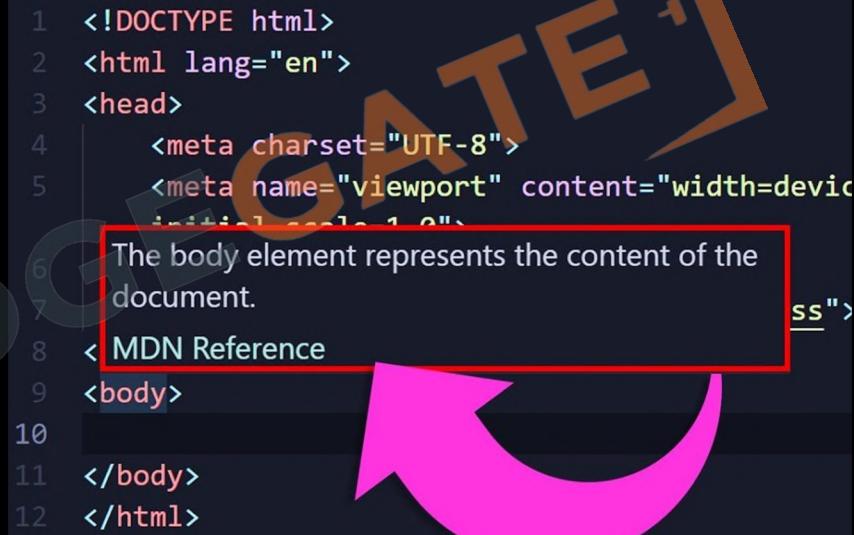
Parent of content tags

Heading tag

# 2.4 MDN Documentation

Tech. Evolve.Only Trustable source of info is Official Documentation Always Up-To-Date

1. Visit <https://developer.mozilla.org/>
2. Official resource for HTML
3. Offers comprehensive guides and tutorials
4. Includes examples for real-world use
5. Updated with latest HTML features
6. Trusted by developers worldwide



The screenshot shows a portion of an MDN documentation page with the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   </head>
7   <body>
8     The body element represents the content of the
9     document.
10    <MDN Reference>
11  </body>
12 </html>
```

A red box highlights the text "The body element represents the content of the document." A large pink arrow points from the bottom right towards this highlighted text.

Hover and click MDN Doc Reference link

# 2.5 Comments

For HUMans,not MACHines

1. Used to add **notes** in HTML code
2. Not displayed on the web page
3. Syntax: `<!-- Comment here -->`
4. Helpful for **code organization**
5. Can be multi-line or single-line

### Writing comments in HTML

**Single-line Comment**

```
1 <!--This is a single line  
comment in HTML. You cannot  
see it on a webpage. Click  
on view-source to see a  
message I left just for you.  
-->
```

**Multi-line Comment**

```
1 <!-- This is a multi-line  
comment in HTML.  
2 You cannot see it on a  
webpage.  
3 If you view-source on the  
browser you can see the  
comment there.-->
```

# 2.6 Case Sensitivity

matlab case sensitive nha  
;) (

1. HTML is case-insensitive for tag names
2. Attribute names are also be case-insensitive
3. Best practice: use lowercase for consistency

<html> = <HTML>  
<p> = <P>  
<head> = <HEAD>  
<body> = <BODY>

best practise

# Level 1 Revision

## HTML Basics

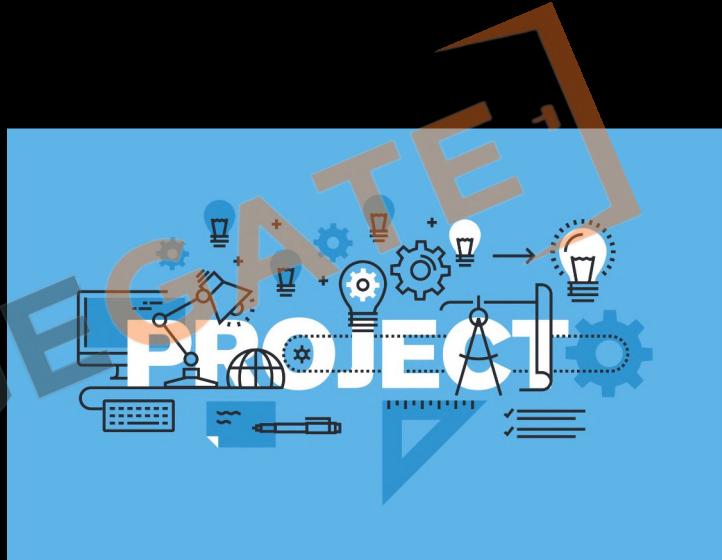
1. Starting up
  1. First File using Text Editor
  2. File Extension
  3. Opening the project in VsCode
  4. Index.html
2. Basics of HTML
  1. What are Tags
  2. Using Emmet ! to generate code
  3. Basic HTML Page
  4. MDN Documentation
  5. Comments
  6. Case Sensitivity



# Project Level 1

## HTML Basics

1. Create a new project with Index.html
2. Generate boilerplate code using Emmet
3. Write “I am learning with Prashant sir”
4. Use comments
5. Also use Case insensitive tags



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Level 2

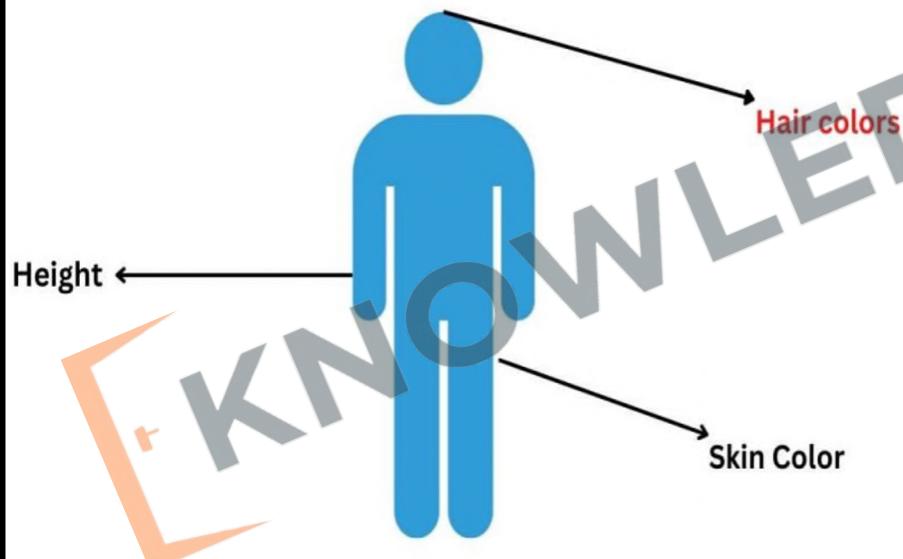
## Must-Use HTML Tags

1. HTML Attributes
  1. What are HTML Attributes
  2. Id Property
2. HTML Tags
  1. Heading Tag
  2. Paragraph Tag
  3. <BR> <HR> tags
  4. Image Tag
  5. Video Tag
  6. Anchor Tag
  7. Bold / Italic / Underline / Strikethrough
  8. Pre Tag
  9. Big / Small Tag
  10. Superscript / Subscript
3. Character Entity Reference
  1. What are Character Entity References

# Level 2

Must-Use HTML Tags

## Attribute



# 1. HTML Attributes

PROPERTIES  
NOT IMP>

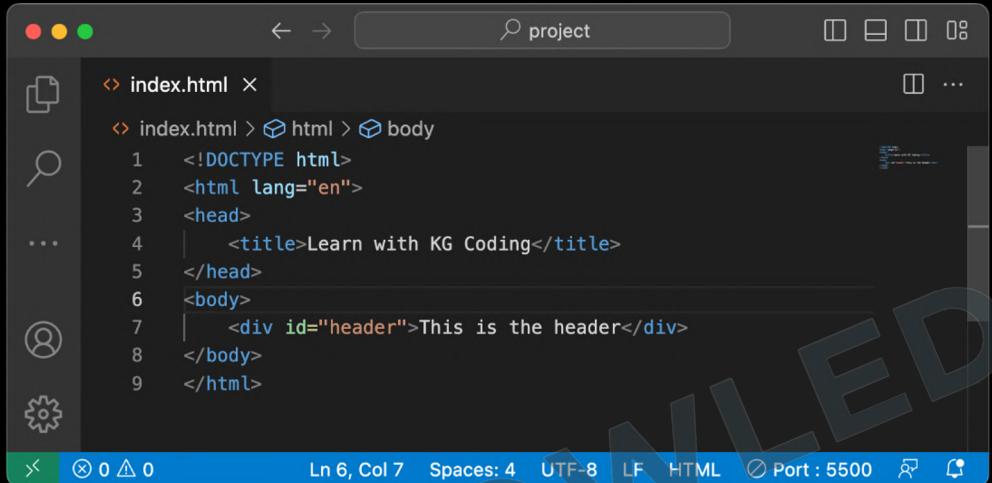
# 1.1 What are HTML Attributes?

## Html Attributes



1. Provides additional information about elements
2. Placed within opening tags
3. Common examples: href, src, alt ALT USED FOR VISUALLY IMPAIRED
4. Use name=value format
5. Can be single or multiple per element

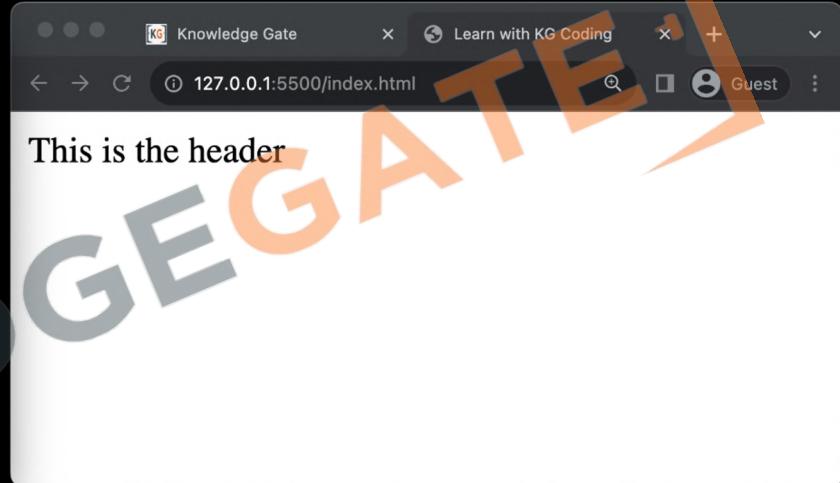
# 1.2 id property



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <div id="header">This is the header</div>
</body>
</html>
```

The code editor interface includes a sidebar with icons for file, search, and settings, and a status bar at the bottom.



- **Unique Identifier:** Each id should be **unique** within a page.
- **Anchoring:** Allows for **direct links to sections** using the **#id** syntax in URLs.
- **CSS & JavaScript:** Used for selecting elements for styling or scripting.

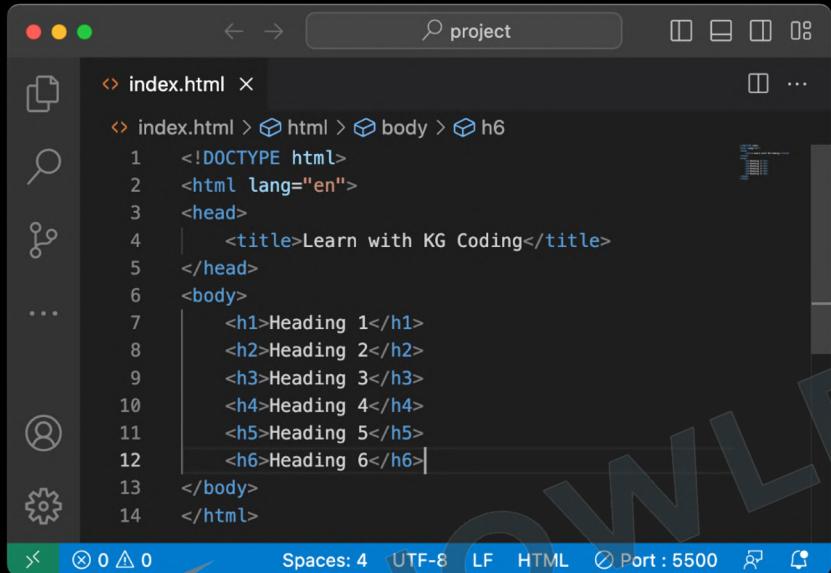
# Level 2

Must-Use HTML Tags



2. HTML  
Tags

# 2.1 Heading Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following HTML structure:

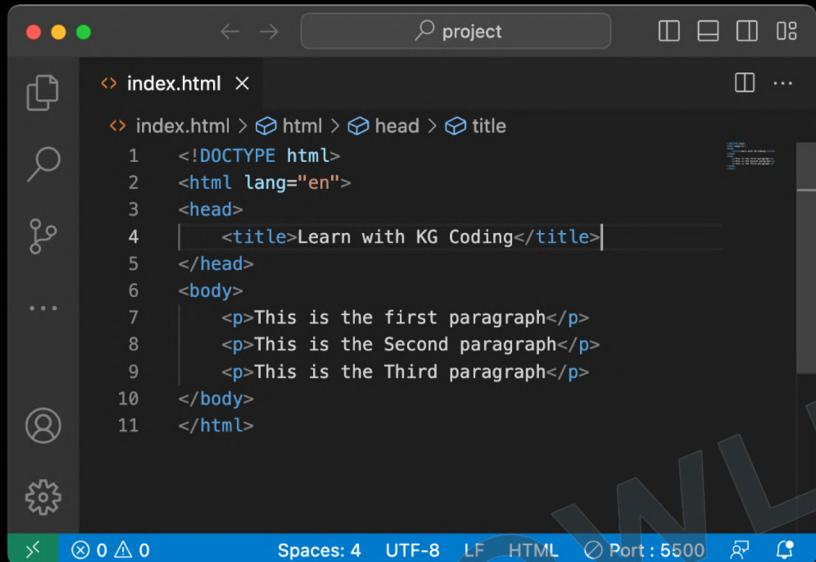
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <h1>Heading 1</h1>
    <h2>Heading 2</h2>
    <h3>Heading 3</h3>
    <h4>Heading 4</h4>
    <h5>Heading 5</h5>
    <h6>Heading 6</h6>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. The status bar at the bottom shows "Spaces: 4", "UTF-8", "LF", "HTML", and "Port : 5500".



1. Defines **headings** in a document
2. Ranges from **<h1>** to **<h6>**
3. **<h1>** is most important, **<h6>** is least
4. Important for SEO
5. Helps in structuring content

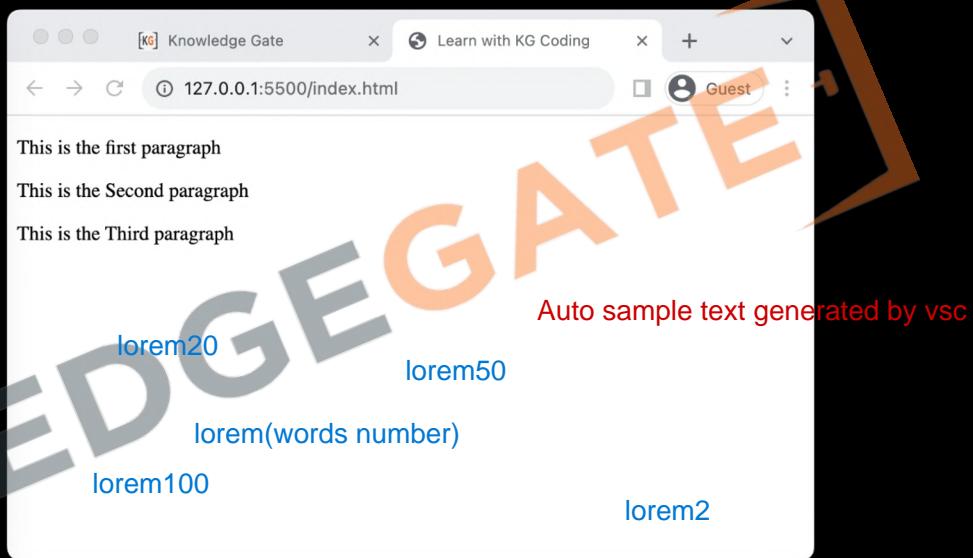
# 2.2 Paragraph Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

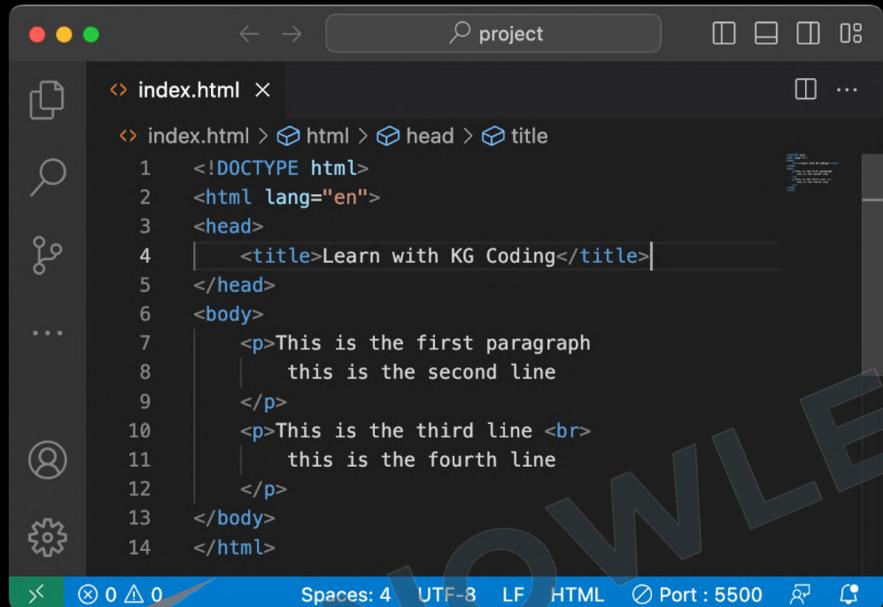
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <p>This is the first paragraph</p>
    <p>This is the Second paragraph</p>
    <p>This is the Third paragraph</p>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. The status bar at the bottom shows "Spaces: 4", "UTF-8", "LF", "HTML", "Port: 5500", and a refresh icon.



1. Used for defining **paragraphs**
2. Enclosed within **<p>** and **</p>** tags
3. Adds **automatic spacing** before and after
4. **Text wraps** to next line inside tag
5. Common in text-heavy content

# 2.3 <BR> Tag



A screenshot of a code editor window titled "index.html". The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <p>This is the first paragraph<br/>
        this is the second line</p>
    <p>This is the third line <br>
        this is the fourth line</p>
</body>
</html>
```

The code editor has a sidebar with various icons for file operations, search, and help.



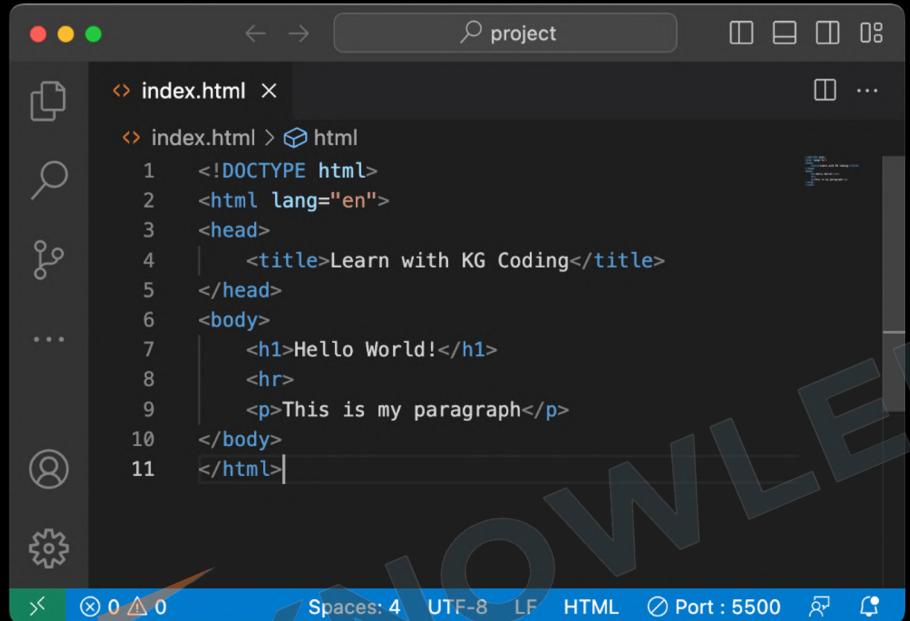
A screenshot of a web browser window titled "Knowledge Gate". The URL is "127.0.0.1:5500/index.html". The content of the page is:

This is the first paragraph this is the second line  
This is the third line  
this is the fourth line

A large, semi-transparent watermark reading "KNOWLEDGE GATE" is overlaid across the bottom of the browser window.

1. <br> adds a line break within text
2. <br> is empty, no closing tag needed
3. <br> and <br /> are both valid

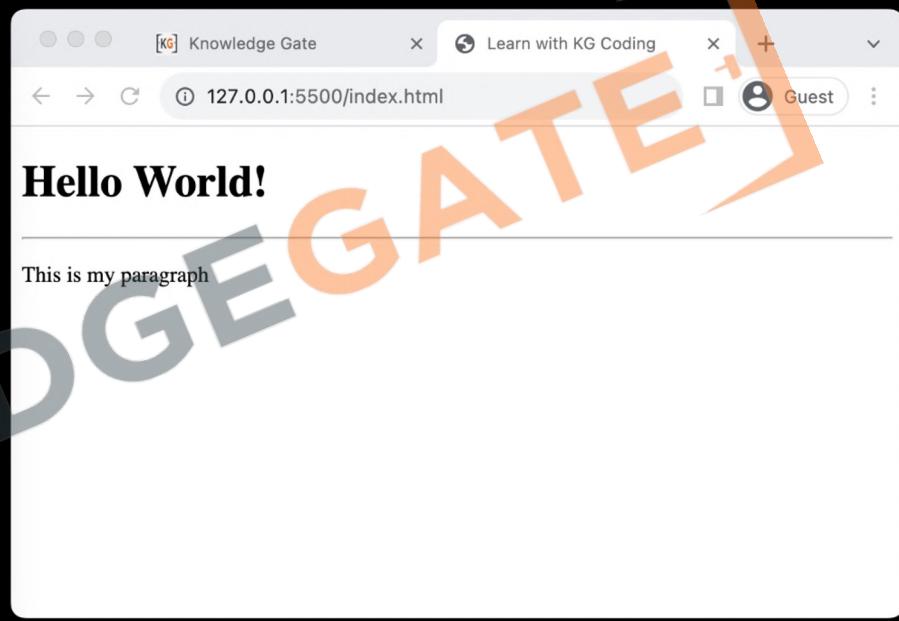
# 2.3 <HR> Tag



A screenshot of a code editor window titled "index.html". The code is as follows:

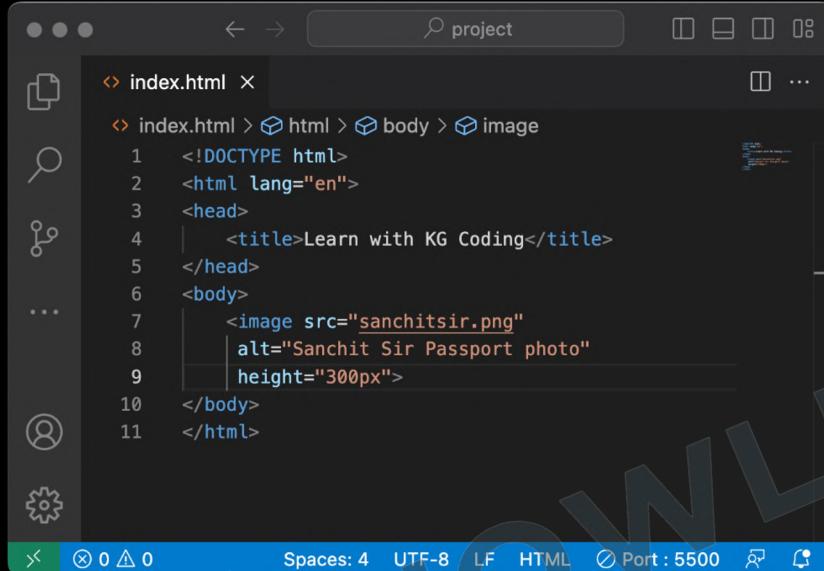
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <h1>Hello World!</h1>
    <hr>
    <p>This is my paragraph</p>
</body>
</html>
```

The code editor has a dark theme with light-colored text. It includes a sidebar with various icons for file operations like copy, paste, and search.



1. <hr> creates a horizontal rule or line
2. <hr> also empty, acts as a divider

# 2.4 Image Tag



A screenshot of a code editor window titled "index.html". The code editor shows the following HTML code:

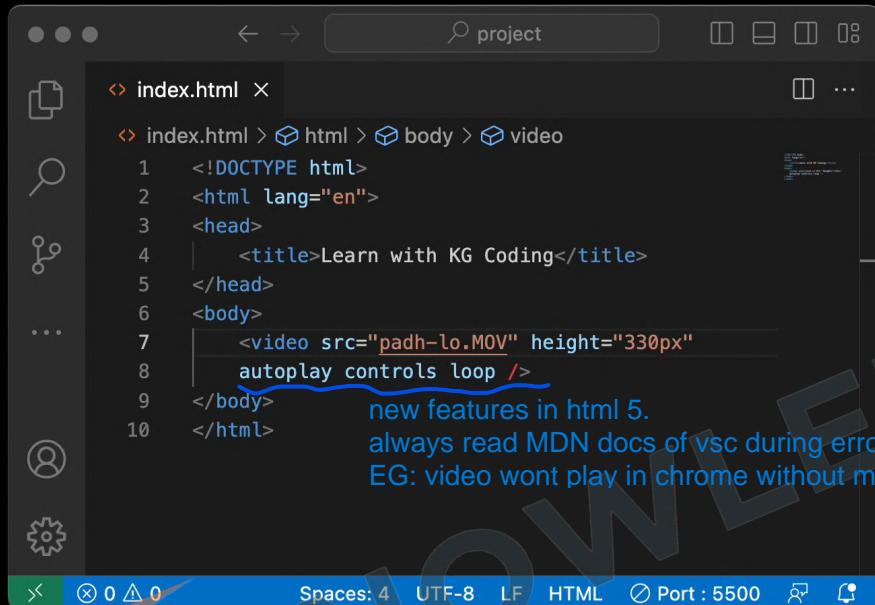
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <image src="sanchitsir.png"
        alt="Sanchit Sir Passport photo"
        height="300px">
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. The sidebar on the left contains various icons for file operations like new, open, save, and search.



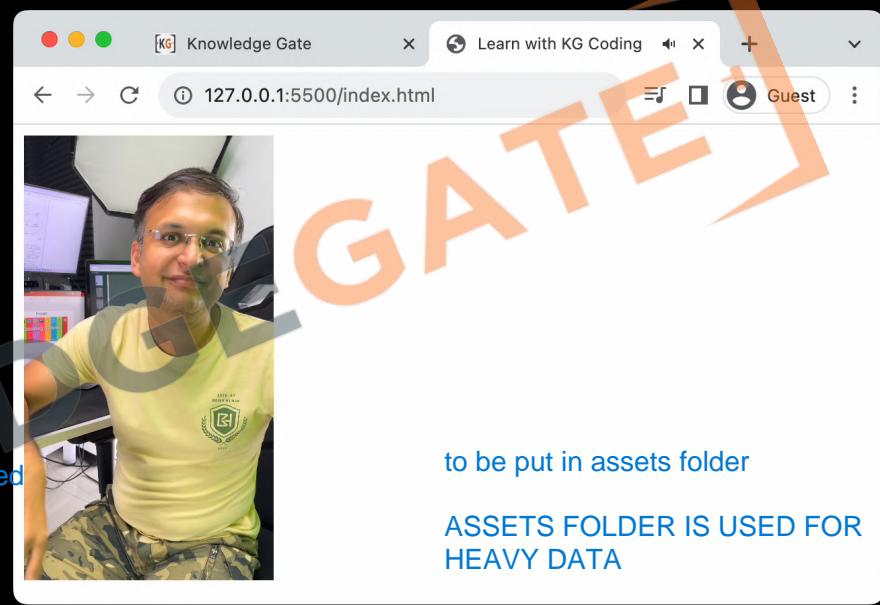
1. Used to embed **images**
2. Utilizes the **src** attribute for image URL ABSOLUTE OR RELATIVE PATH OR ONLINE LINK
3. **alt** attribute for alternative text ACCESSIBILITY OPTION
4. Can be resized using **width** and **height** ONLY ONE SHOULD BE USED TO MAINTAIN ASPECT RATIO
5. **Self-closing**, doesn't require an end tag

# 2.5 Video Tag



```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <video src="padh-lo.MOV" height="330px"
        autoplay controls loop />
</body>
</html>
```

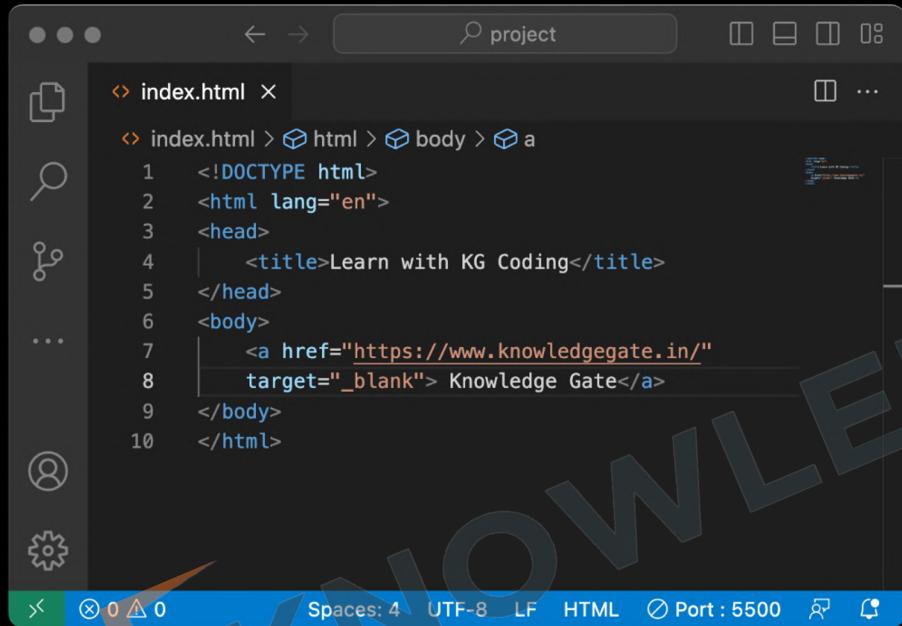
new features in html 5.  
always read MDN docs of vsc during error  
EG: video wont play in chrome without muted



1. Embeds video files on a page
2. Uses **src** attribute for video URL
3. Supports **multiple formats** like MP4, WebM
4. Allows for built-in controls via attributes like **autoplay, controls, loop**

# 2.6 Anchor Tag

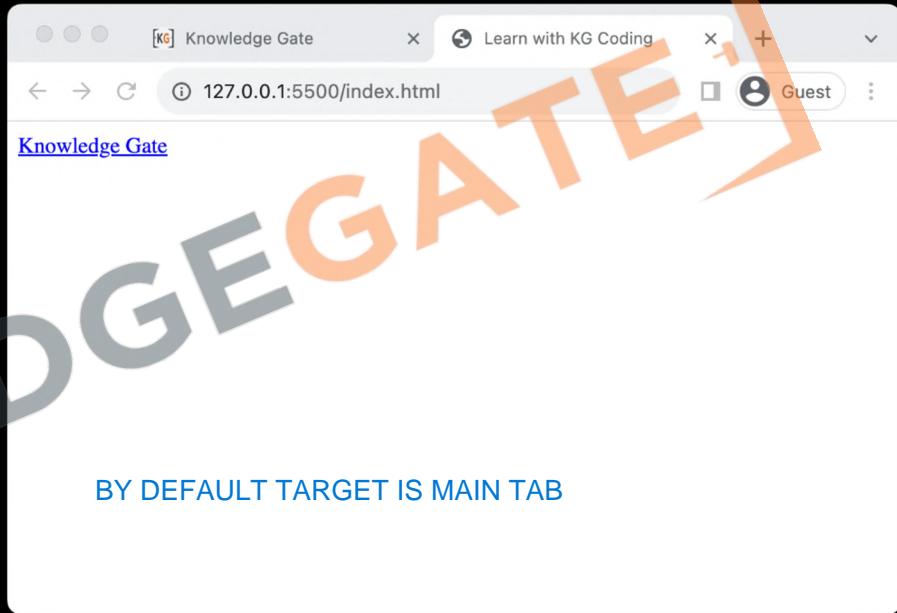
LINK TO ANYTHING ELSE



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

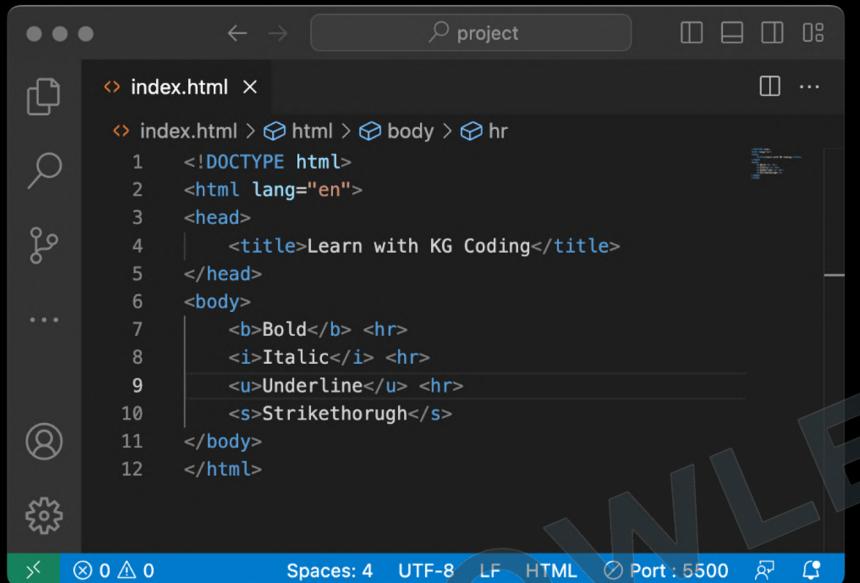
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <a href="https://www.knowledgegate.in/" target="_blank"> Knowledge Gate</a>
</body>
</html>
```

The line containing the anchor tag is highlighted.



1. Used for creating **hyperlinks**
2. Requires **href** attribute for URL
3. Can link to **external sites or internal pages**
4. Supports **target** attribute to control link behavior

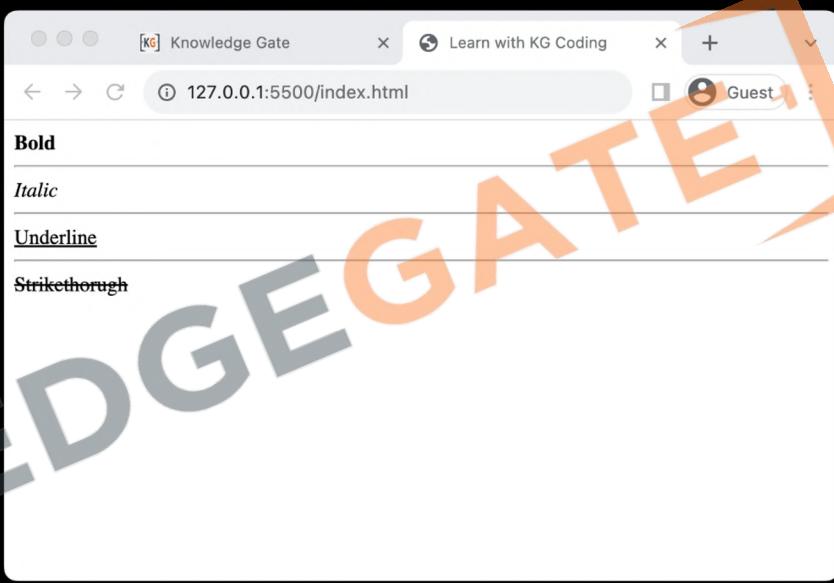
# 2.7 Bold/Italic/Underline/Strikethrough Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, displaying the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <b>Bold</b> <hr>
    <i>Italic</i> <hr>
    <u>Underline</u> <hr>
    <s>Strikethrough</s>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. It includes a sidebar with various icons and a bottom status bar showing "Spaces: 4", "UTF-8", "LF", "HTML", "Port : 5500", and a refresh icon.



1. **<b>** makes text bold
2. **<i>** makes text italic
3. **<u>** underlines text
4. **<s>** or **<strike>** applies strikethrough
5. Primarily used for text styling and emphasis

# 2.8 Pre Tag

A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

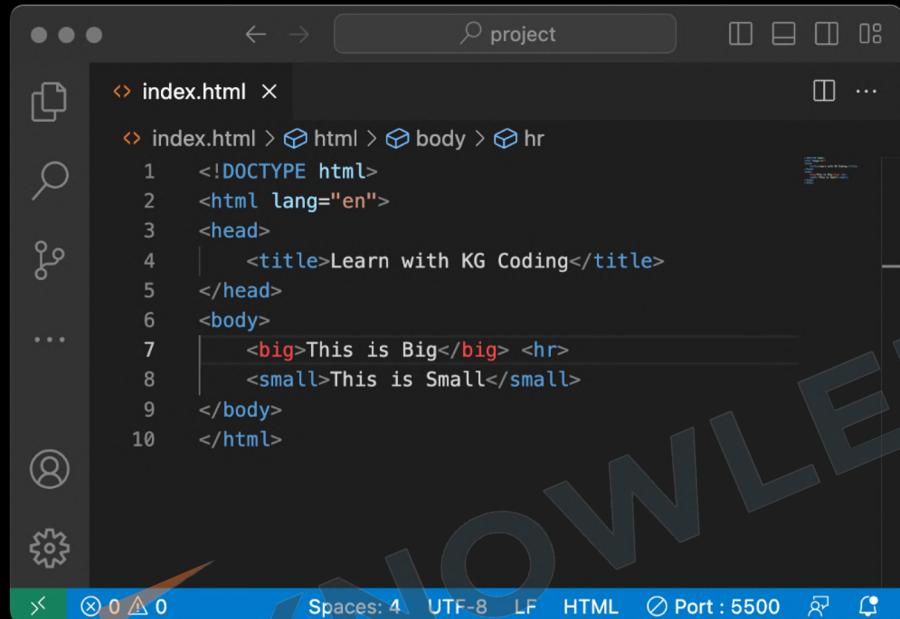
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Learn with KG Coding</title>
5 </head>
6 <body>
7   first line      PREFORMATTED STUFF
8   more    spaces
9   <pre>
10  first line
11  more    spaces
12 </pre>
13 </body>
14 </html>
```

The code editor has a dark theme with light-colored syntax highlighting. A large watermark reading "KNOWLEDGE GATE" is diagonally across the screen.



1. Preserves text formatting
2. Maintains whitespace and line breaks
3. Useful for displaying code
4. Enclosed within `<pre>` and `</pre>` tags

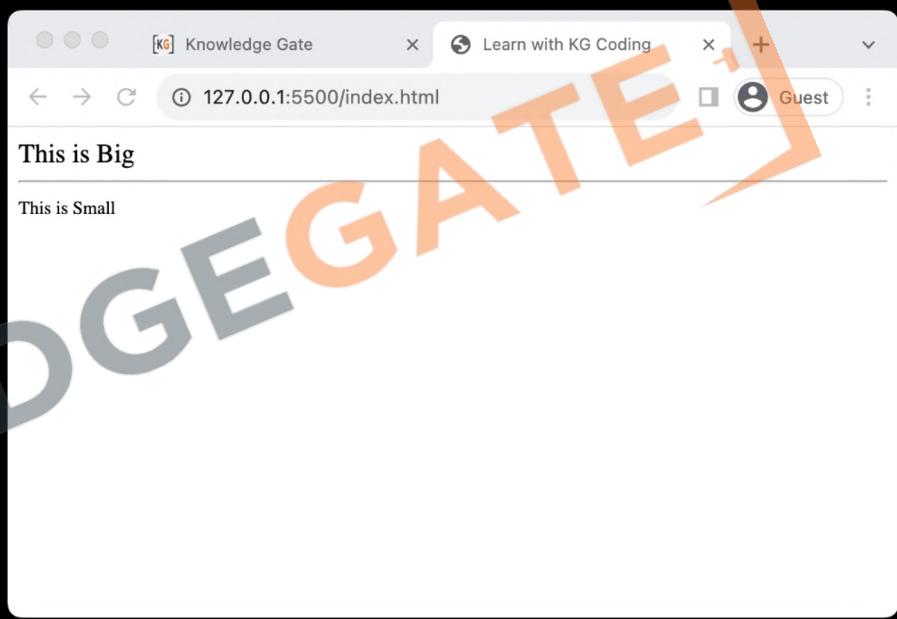
# 2.9 Big/Small Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <big>This is Big</big> <hr>
    <small>This is Small</small>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. A sidebar on the left contains various icons for file operations like copy, paste, search, and user management.



1. `<big>` increases text size
2. `<small>` decreases text size
3. Less common due to CSS alternatives

# 2.10 Superscript/Subscript Tag

A screenshot of a code editor window titled "index.html". The code is as follows:

```
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <big>(a + b)2 = a2 + b2 + 2ab</big> <hr>
    <big>(a + b)2 = a2 + b2 + 2ab</big> +
    b2 + 2ab</big> <hr>

    <big>CH4 + O2 => H2O + CO2</big> <hr>
    <big>CH4 + O2 => H2O + CO2</big> <hr>
</body>
</html>
```

A screenshot of a web browser window titled "Learn with KG Coding". The URL is 127.0.0.1:5500/index.html. The page displays the following mathematical equations:

$$(a + b)^2 = a^2 + b^2 + 2ab$$
$$(a + b)^2 = a^2 + b^2 + 2ab$$
$$\text{CH}_4 + \text{O}_2 \Rightarrow \text{H}_2\text{O} + \text{CO}_2$$
$$\text{CH}_4 + \text{O}_2 \Rightarrow \text{H}_2\text{O} + \text{CO}_2$$

1. `<sup>` makes text superscript
2. `<sub>` makes text subscript
3. Used for mathematical equations, footnotes
4. Does not change font size, just position

# Level 2

Must-Use HTML Tags

## 3. Character Entity Reference

Reference

# 3.1 Character Entity Reference

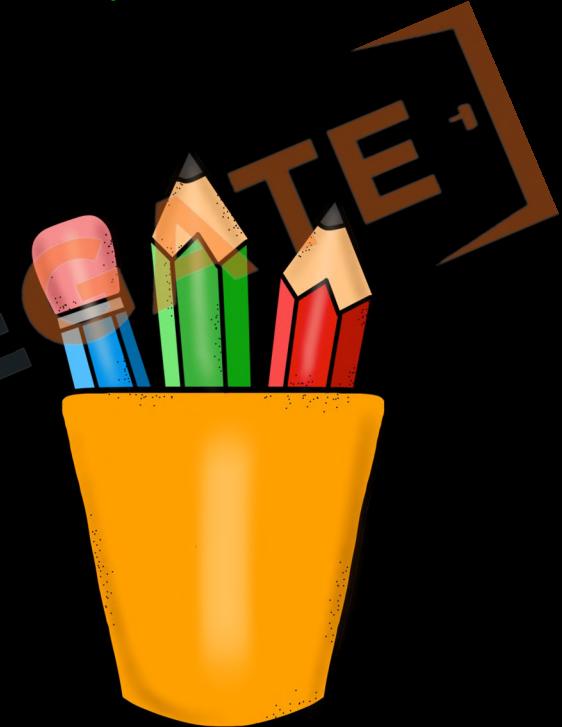
1. Used to display reserved or special characters
2. Syntax often starts with & and ends with w (e.g., &amp; for &)

&nbsp;	-	&ndash;	-	&minus;	°	&deg;	Δ	&Delta;	α	&alpha;	
€	&euro;	—	&mdash;	±	&plusmn;	°	&ordm;	Λ	&Lambda;	β	&beta;
ƒ	&cent;	…	&hellip;	√	&radic;	ª	&ordf;	Θ	&Theta;	γ	&gamma;
£	&pound;	§	&sect;	∞	&infin;	¹	&sup1;	Ξ	&Xi;	δ	&delta;
¥	&yen;	¶	&para;	∞	&prop;	²	&sup2;	Π	&Pi;	ε	&epsilon;
¤	&curren;	†	&dagger;	×	&times;	³	&sup3;	Σ	&Sigma;	ζ	&zeta;
f	&fnof;	‡	&Dagger;	÷	&divide;	¼	&frac14;	Φ	&Phi;	η	&eta;
©	&copy;	®	&reg;	~	&sim;	½	&frac12;	Ψ	&Psi;	θ	&theta;
®	&reg;	®	&iquest;	≈	&asymp;	¾	&frac34;	Ω	&Omega;	ι	&iota;
™	&trade;	%	&permil;	≡	&cong;	∴	&there4;	∇	&nabla;	κ	&kappa;

# Level 2 Revision

## Must-Use HTML Tags

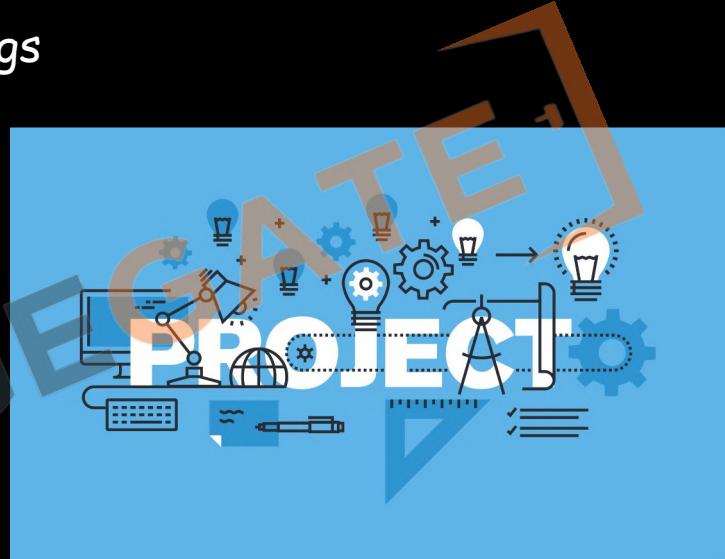
1. HTML Attributes
  1. What are HTML Attributes
  2. Id Property
2. HTML Tags
  1. Heading Tag
  2. Paragraph Tag
  3. <BR> <HR> tags
  4. Image Tag
  5. Video Tag
  6. Anchor Tag
  7. Bold / Italic / Underline / Strikethrough
  8. Pre Tag
  9. Big / Small Tag
  10. Superscript / Subscript
3. Character Entity Reference
  1. What are Character Entity References



# Project Level 2

Must-Use HTML Tags

1. Create a page with **heading**, **paragraph**, **line breaks** and **separators**.
2. Use an **image** with height 300, which is a **link** to another page.
3. Use **bold**, **italic**, **underline** and **strike through** in one line.
4. Write third equation of motion using **superscript** and **subscript**.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Level 3

## Browser Tools

1. Browser Tools
  1. View Page Source
  2. Inspect Element
  3. HTML without CSS
2. Responsive Design
  1. Different screen size
3. Live Edit Code
  1. Live edit HTML
  2. Live edit CSS
  3. Live edit JS
  4. Changes only happening at client
4. Validating Web pages
  1. Using validator.w3.org

# Level 3

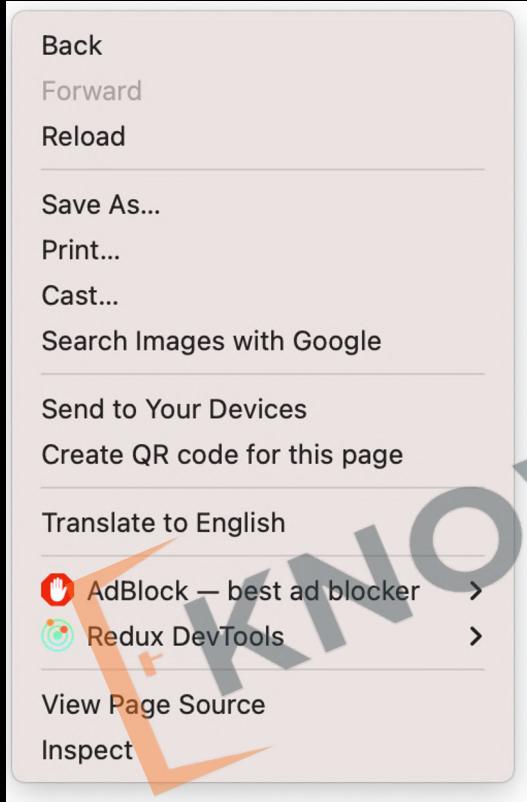
## Browser Tools



# 1. Browser Tools

# 1.1 View Page Source

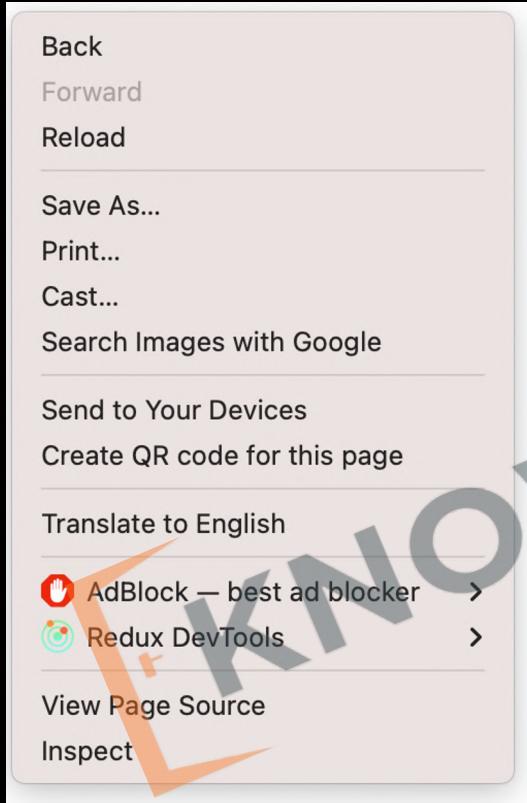
or view-source:<https://google.com>



1. Displays raw HTML and CSS
2. Useful for debugging and learning
3. Shows external files like JavaScript links

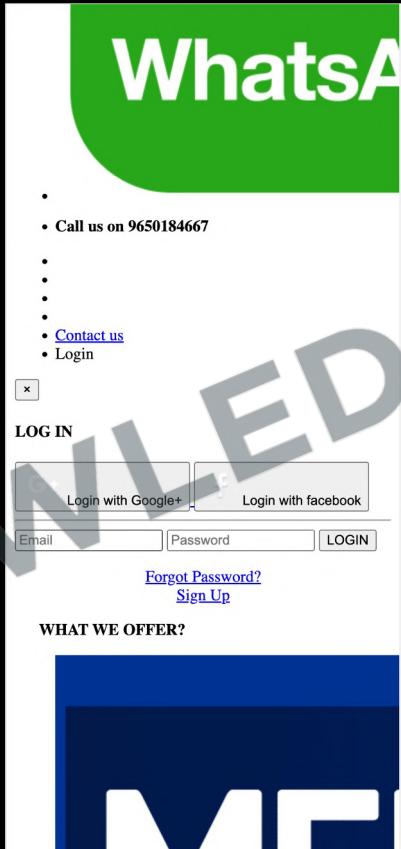
inspect webpage or inspect element

# 1.2 Inspect Element

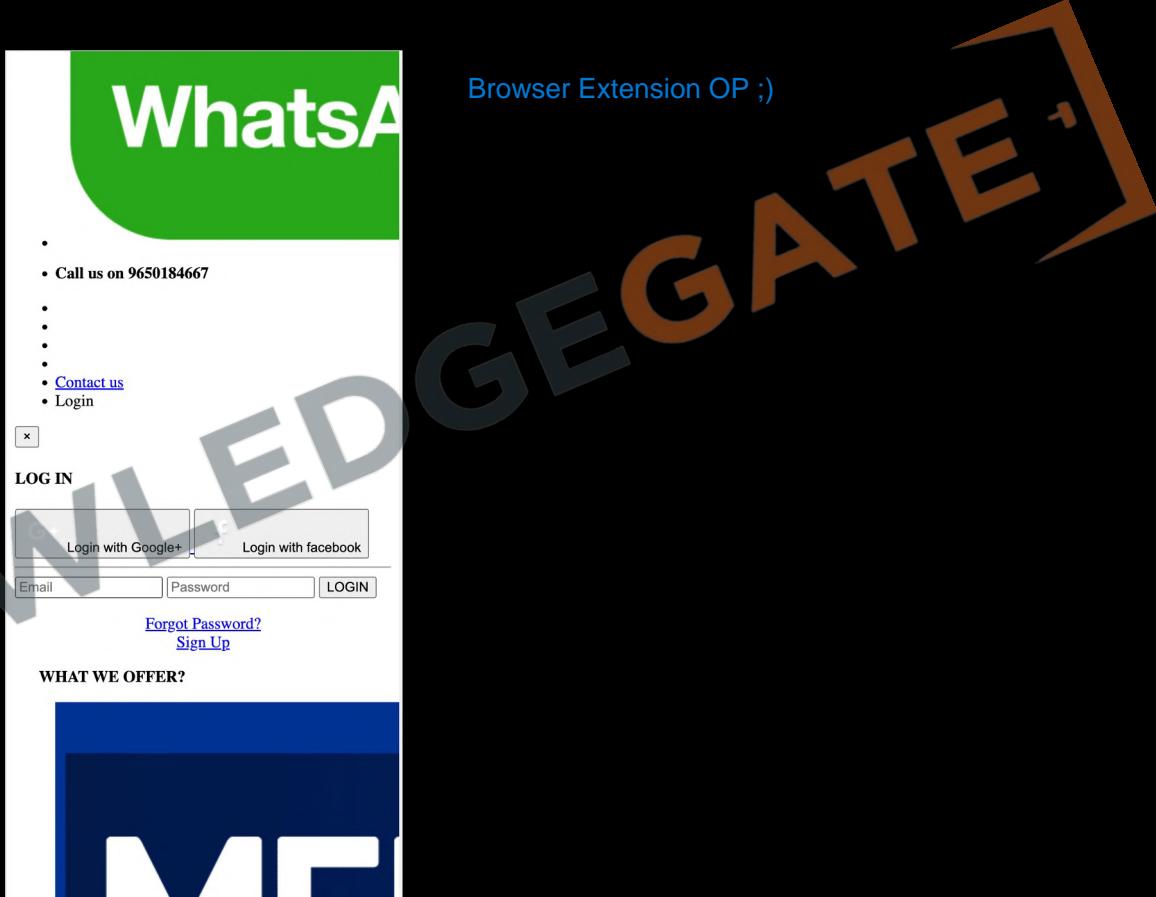


1. Allows **real-time editing** of HTML/CSS
2. Useful for **debugging** and testing
3. Shows **element hierarchy** and layout
4. Includes console for **JavaScript**
5. **Highlights** selected elements on page

# 1.3 HTML without CSS



Browser Extension OP ;)



# Level 3

## Browser Tools



KNOWLEDGE GATE<sup>1</sup>  
**Responsive  
Design**

# 2.1 Different Screen Sizes



1. Adapts layout for different screen sizes
2. Flexible layouts
3. Optimizes images and assets
4. Enhances user experience on mobile and desktop

Inspect > Elements > Dimensions :

# Level 3

Browser Tools



3.GATE<sup>1</sup>  
Live Edit  
Code

# 3.1 Live edit HTML

YouTube IN

kg coding

**KG Coding**

@KG\_Coding 10M subscribers 1 video

Coming Soon..... >

[knowledgegate.in](http://knowledgegate.in) and 1 more link

**HOME**   **VIDEOS**   **LIVE**   **PLAYLISTS**

**Our Other Channels**

**KNOWLEDGE GATE**  
594K subscribers

**KnowledgeGate Placement Prep**  
105K subscribers

Subscribed

Subscribe

```
> renderer>
  > ytd-channel-name id="channel-name" wrap-text class="style-scope ytd-c4-tabbed-header-renderer"> ...
  </ytd-channel-name>
  > sp class="style-scope ytd-c4-tabbed-header~renderer">
    > span class="meta-item style-scope ytd-c4-tabbed-header~renderer"> ...
    > span class="meta-item style-scope ytd-c4-tabbed-header~renderer" hidden="true" ...> ...
    > span class="meta-item style-scope ytd-c4-tabbed-header~renderer">
      > yt-formatted-string id="subscriber-count" class="style-scope ytd-c4-tabbed-header~renderer" aria-label="13.8K subscribers" 10M subscribers</yt-formatted-string> == $0
      <span aria-hidden="true" class="delimiter style-scope ytd-c4-tabbed-header~renderer"> ...</span>
    > span class="meta-item style-scope ytd-c4-tabbed-header~renderer"> ...
    > div id="channel-tagline" class="style-scope ytd-c4-tabbed-header~renderer"> ...
    > div id="channel-header-links" class="style-scope ytd-c4-tabbed-header~renderer"> ...
    > div id="buttons" class="style-scope ytd-c4-tabbed-header~renderer"> ...
  </div>
  > div id="links-holder" class="style-scope ytd-c4-tabbed-header~renderer"> ...
  </div>
  </div>
  <tp-yt-app-toolbar sticky class="style-scope ytd-c4-tabbed-header~renderer" style="transform: translate3d(0px, 0px, 0px); "> ...
  </tp-yt-app-toolbar> <div> ...
  </div>
</tp-yt-app-header>
<div id="contentContainer" class="style-scope tp-yt-app-header-layout" style="padding-top: 364px;"></div>
</div>
</tp-yt-app-header-layout>
</ytd-c4-tabbed-header-renderer>
</div>
<div id="alerts" class="style-scope ytd-browse"></div>
<ytd-channel-legal-info-renderer class="style-scope ytd-browse" disable-upgrade hidden="true" ></ytd-channel-legal-info-renderer>
<ytd-playlist-sidebar-renderer class="style-scope ytd-browse" disable-upgrade hidden="true" ></ytd-playlist-sidebar-renderer>
<ytd-playlist-header-renderer class="style-scope ytd-browse" disable-upgrade hidden="true" ></ytd-playlist-header-renderer>
<ytd-settings-sidebar-renderer class="style-scope ytd-browse" disable-upgrade hidden="true" ></ytd-settings-sidebar-renderer>
<ytd-two-column-browse-results-renderer class="style-scope ytd-browse" id="grid-4-columns" page-subtype="channels" style="touch-action: pan-y;">
  <div id="primary" class="style-scope ytd-section-list-renderer">
    <tp-yt-section-list-renderer class="style-scope ytd-section-list-renderer">
      <div id="section-item" class="style-scope ytd-section-item-renderer" page-subtype="channels"> ...
        <div id="header-container" class="style-scope ytd-section-list-renderer">
          <div id="header"> ...
            <div id="header-content" class="style-scope ytd-section-item-renderer"> ...
              <div id="header-content"> ...
                <div id="header-content"> ...
                  <div id="header-content"> ...
                    <div id="header-content"> ...
                      <div id="header-content"> ...
                        <div id="header-content"> ...
                          <div id="header-content"> ...
                            <div id="header-content"> ...
                              <div id="header-content"> ...
                                <div id="header-content"> ...
                                  <div id="header-content"> ...
                                    <div id="header-content"> ...
                                      <div id="header-content"> ...
                                        <div id="header-content"> ...
                                          <div id="header-content"> ...
                                            <div id="header-content"> ...
                                              <div id="header-content"> ...
                                                <div id="header-content"> ...
                                                  <div id="header-content"> ...
                                                    <div id="header-content"> ...
                                                      <div id="header-content"> ...
                                                        <div id="header-content"> ...
                                                          <div id="header-content"> ...
                                                            <div id="header-content"> ...
                                                              <div id="header-content"> ...
                                                                <div id="header-content"> ...
                                                                  <div id="header-content"> ...
                                                                    <div id="header-content"> ...
                                                                      <div id="header-content"> ...
                                                                        <div id="header-content"> ...
                                                                          <div id="header-content"> ...
                                                                            <div id="header-content"> ...
                                                                              <div id="header-content"> ...
                                                                                <div id="header-content"> ...
                                                                                  <div id="header-content"> ...
                                                                                    <div id="header-content"> ...
                                                                                      <div id="header-content"> ...
                                                                                        <div id="header-content"> ...
                                                                                          <div id="header-content"> ...
                                                                                            <div id="header-content"> ...
                                                                                              <div id="header-content"> ...
                                                                                                <div id="header-content"> ...
                                                                                                  <div id="header-content"> ...
                                                                                                    <div id="header-content"> ...
                                                                                                     ...

```

Changed Subscriber count

# 3.2 Live edit CSS

The screenshot shows the KG Coding channel page on YouTube. The channel name 'kg coding' is visible in the search bar. The channel profile picture is a portrait of a man with glasses. The channel title is 'KG Coding' with 10M subscribers and 1 video. A 'Coming Soon..... >' message is displayed. Below the main content, there are sections for 'HOME', 'VIDEOS', 'LIVE', and 'PLAYLISTS'. A large orange banner at the bottom features the text 'CODING MADE' and logos for Angular, Vue.js, MERN, Express, Node.js, C, C++, and Java. Two other channels are listed under 'Our Other Channels': 'KNOWLEDGE GATE' (594K subscribers) and 'KnowledgeGate Placement Prep' (105K subscribers). A 'Subscribe' button is at the bottom.

```
<div>
  <yt-c4-channel-name id="channel-name" wrap-text class="style-scope ytd-c4-tabbed-header-renderer">
    <!--css-build:shady-->
    <!--css-build:shady-->
    <div id="container" class="style-scope ytd-channel-name">
      <yt-formatted-string id="text" link-inherit-color title class="style-scope ytd-channel-name" e="KG Coding-</yt-formatted-string> =&gt;
    </div>
    <tp-yt-paper-tooltip fit-to-visible-bounds class="style-scope ytd-channel-name" role="tooltip" ip="" tabindex="-1"></tp-yt-paper-tooltip>
  </div>
  <ytd-badge-supported-renderer class="style-scope ytd-channel-name" disable-upgrade hidden></ytd-badge-supported-renderer>
</yt-c4-channel-name>
<p class="style-scope ytd-c4-tabbed-header-renderer" hidden></p>
<span class="meta-item style-scope ytd-c4-tabbed-header-renderer"></span>
<span class="meta-item style-scope ytd-c4-tabbed-header-renderer" hidden></span>
<span class="meta-item style-scope ytd-c4-tabbed-header-renderer">
  <yt-formatted-string id="subscriber-count" class="style-scope ytd-c4-tabbed-header-renderer" aria-label="13.8K subscribers">10M subscribers</yt-formatted-string>
  <span aria-hidden="true" class="delimiter style-scope ytd-c4-tabbed-header-renderer"></span>
</span>
<span class="meta-item style-scope ytd-c4-tabbed-header-renderer"></span>
<div id="channel-tagline" class="style-scope ytd-c4-tabbed-header-renderer"></div>
<div id="channel-header-links" class="style-scope ytd-c4-tabbed-header-renderer"></div>
</div>
<div id="buttons" class="style-scope ytd-c4-tabbed-header-renderer"></div>
</div>
</div>
<div id="links-holder" class="style-scope ytd-c4-tabbed-header-renderer"></div>
</div>
</div>
<tp-yt-app-toolbar sticky class="style-scope ytd-c4-tabbed-header-renderer" style="transform: translate3d(0px, 0px, 0px); "></tp-yt-app-toolbar>
</div>
</tp-yt-app-header>
<div id="contentContainer" class="style-scope tp-yt-app-header-layout" style="padding-top: 364px;"></div>
</div>
</tp-yt-app-header-layout>
</yt-c4-tabbed-header-renderer>
</div>
<div id="alerts" class="style-scope ytd-browse"></div>
<ytd-channel-legal-info-renderer class="style-scope ytd-browse" disable-upgrade hidden></ytd-channel-legal-info-renderer>
<ytd-playlist-sidebar-renderer class="style-scope ytd-browse" disable-upgrade hidden></ytd-playlist-sidebar-renderer>
<div>
```

Changed Channel Name color

# 3.3 Live edit JS

The image shows a YouTube channel interface for 'KONLEGATE'. On the left, there's a sidebar with icons for Home, Shorts, Subscriptions, and Library. The main area features a video thumbnail for a video titled 'Coding Made Simple' which includes logos for Angular, Vue.js, MongoDB, Express, React, and Node.js. Below the video thumbnail are navigation links for HOME, VIDEOS, LIVE, and PLAYLISTS.

On the right side of the image, a DevTools console window is open, displaying a list of errors. The errors are as follows:

- com/s/desktop/462a8050/sjoin/custom-elements-ess-adapt...@k8-opt/bin/cnird\_party.js: vascript/custom\_elements/fast-shim.js.sourcemap: HTTP error: status code 404, net::ERR\_HTTP\_RESPONSE\_CODE\_FAILURE
- DevTools failed to load source map: Could not load content for https://www.youtube.com/s/desktop/462a8d5d/jstab/web-animations-next-lite.min.vfset/web-animations-next-lite.min.js.map: HTTP error: status code 404, net::ERR\_HTTP\_RESPONSE\_CODE\_FAILURE
- DevTools failed to load source map: Could not load content for https://www.youtube.com/s/desktop/462a8d5d/jstab/webcomponents-sd.vfset/bl...\_party/javascript/polymer/v2/webcomponentsjs/webcomponents-sd.js.sourcemap: HTTP error: status code 404, net::ERR\_HTTP\_RESPONSE\_CODE\_FAILURE
- LegacyDataMixin will be applied to all desktop\_polymer\_enable\_wl\_icons.js:4482 legacy elements.
- Set '\_legacyUndefinedCheck: true' on element class to enable.
- GET https://googleads.g.doubleclick.net/pagead/id net::ERR\_BLOCKED\_BY\_CLIENT VM15538:201
- GET https://www.google.com/pagead/lv www.google.com/pagea...5CoWhRv 7RpGsmwVg:1 z?evtid=ACd6KtzZ40h0LZgbnYYZ8gxW6cxP BX2-4..inAppBootstrap%3AUnclassified&az=1&sigh=AB9vU40PfuZ9Fiv1V5CoWhRv 7RpGsmwVg net::ERR\_BLOCKED\_BY\_CLIENT
- GET https://www.google.co.in/pagead/ www.google.co.in/pag...5CoWhRv 7RpGsmwVg:1 lvz?evtid=ACd6KtzZ40h0LZgbnYYZ8gxW6cxP xPBX2..inAppBootstrap%3AUnclassified&az=1&sigh=AB9vU40PfuZ9Fiv1V5CoWhRv 7RpGsmwVg net::ERR\_BLOCKED\_BY\_CLIENT
- chrome-extension://invalid/ net::ERR\_FAILED cast sender.js:10
- chrome-extension://invalid/ net::ERR\_FAILED cast sender.js:10
- GET https://static.doubleclick.net/instream/ad\_status.js net::ERR\_BLOCKED\_BY\_CLIENT spf.js:34
- document.getElementById("inner-header-container").style.visibility = 'hidden'  
< 'hidden'
- The resource https://i.ytimg.com/generate\_204 was preloaded using link featured:1 preload but not used within a few seconds from the window's load event. Please make sure it has an appropriate 'as' value and it is preloaded intentionally.

# 3.4 Changes happening at Client

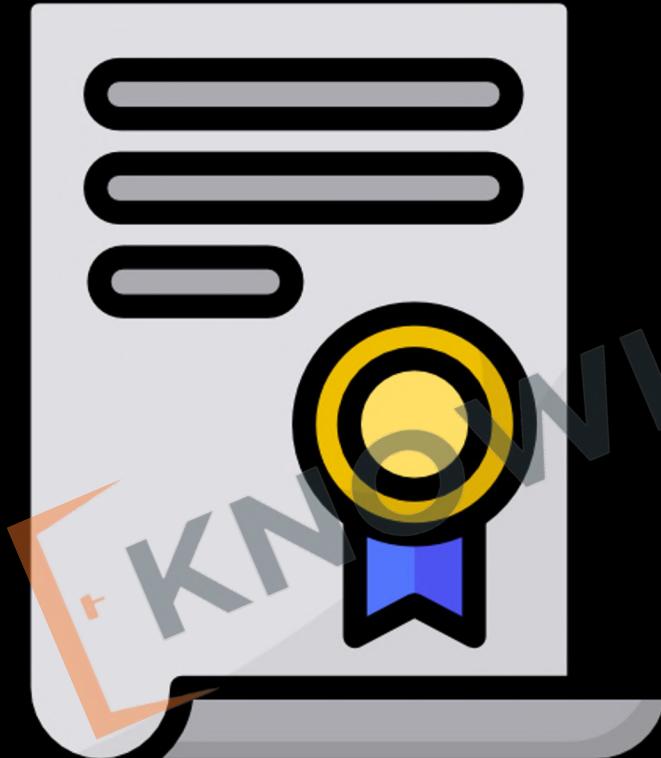
1. Changes made are **temporary**
2. Affect **only** the current session
3. Not saved to the server not at master copy
4. Reset upon page **reload**
5. Useful for **testing**, not permanent fixes



Like: If you change the question in your question paper that has no effect on actual exam.

# Level 3

## Browser Tools



# 4. Validating WebPages

In these time,Modern Browsers are soo smart that they forgive little syntax errors

# 4.1 Using validator.w3.org

to actually check syntax we use [validator.w3.org](https://validator.w3.org)

Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

Showing results for contents of text-input area

Checker Input

Show  source  outline  image report Options...

Check by  text input  css

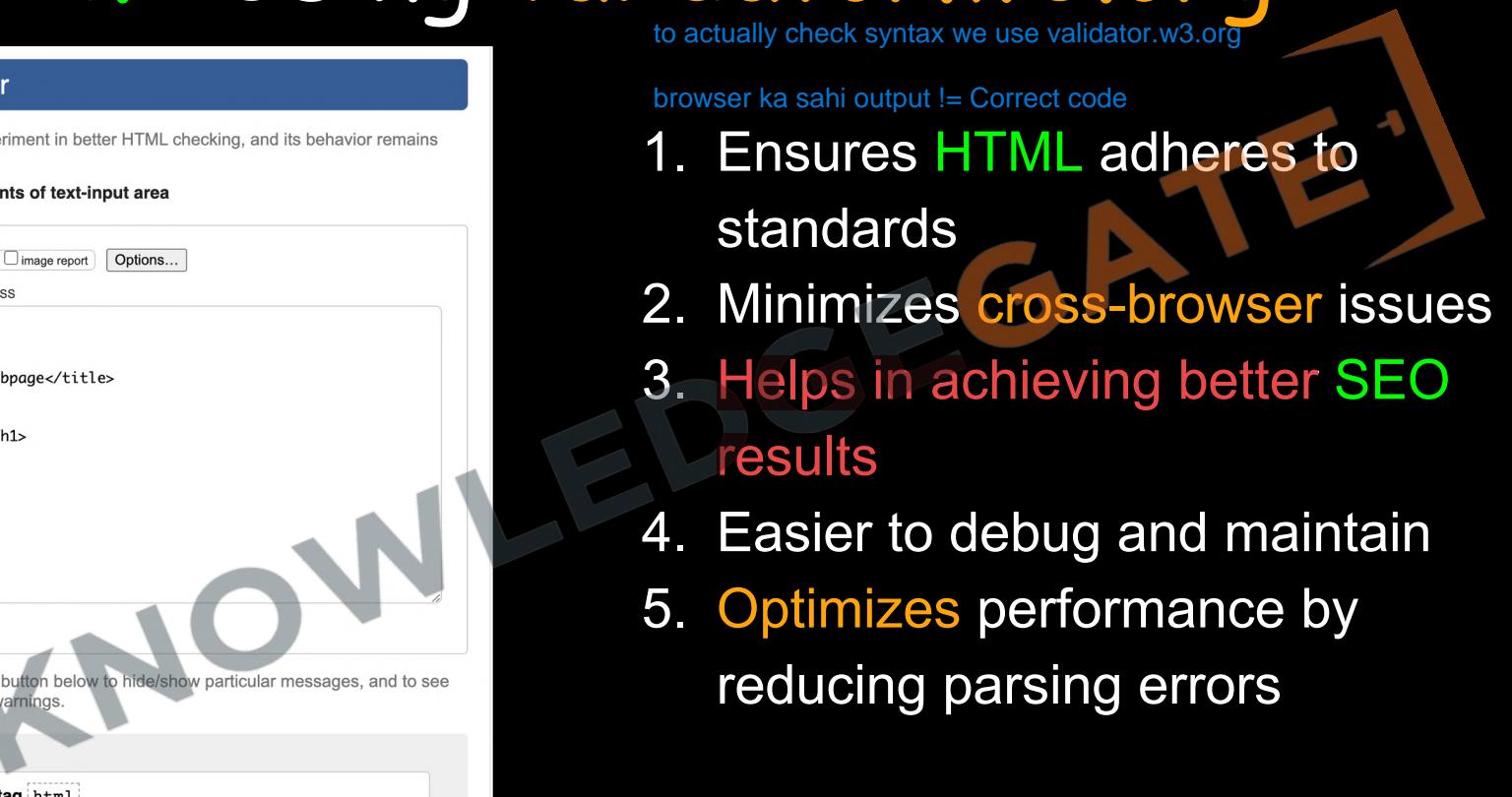
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>My First Webpage</title>
</head>
<body>
    <h1>Hello World!</h1>
</body>
<html>
```

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

Message Filtering

1. Error Stray start tag `html`.  
From line 9, column 1; to line 9, column 6  
`></body><html>`

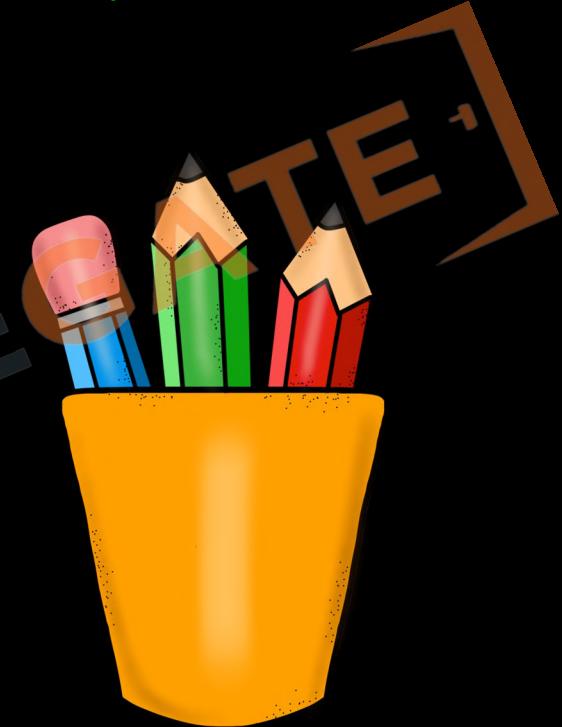


1. Ensures **HTML** adheres to standards
2. Minimizes cross-browser issues
3. Helps in achieving better **SEO** results
4. Easier to debug and maintain
5. Optimizes performance by reducing parsing errors

# Level 3 Revision

## Browser Tools

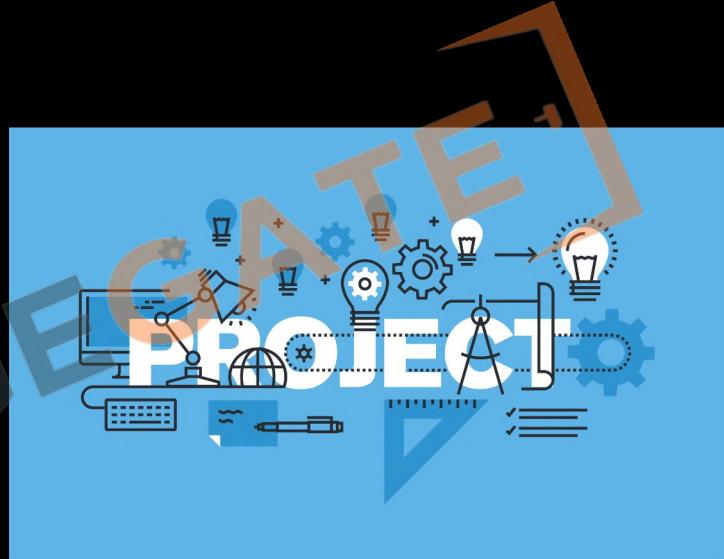
1. Browser Tools
  1. View Page Source
  2. Inspect Element
  3. HTML without CSS
2. Responsive Design
  1. Different screen size
3. Live Edit Code
  1. Live edit HTML
  2. Live edit CSS
  3. Live edit JS
  4. Changes only happening at client
4. Validating Web pages
  1. Using validator.w3.org



# Project Level 3

## Browser Tools

1. Save Source of **Instagram** in a file and check the render.
2. **Inspect** the likes element on the page and read the code to understand.
3. Change number of likes on Your **Instagram** post
4. **Validate** the page we created in last project.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Level 4

## HTML and Project Structure

1. Semantic Tags
  1. Semantic / Non-Semantic Tags
2. Body Tags
  1. Header Tag
  2. Main Tag
    1. Section Tag
    2. Article Tag
    3. Aside Tag
  3. Footer Tag
3. Folder Structure
  1. Recommended Folder structure
4. More Tags
  1. Navigation tags
  2. Block / Inline Elements
  3. Div tags
  4. Span Tags

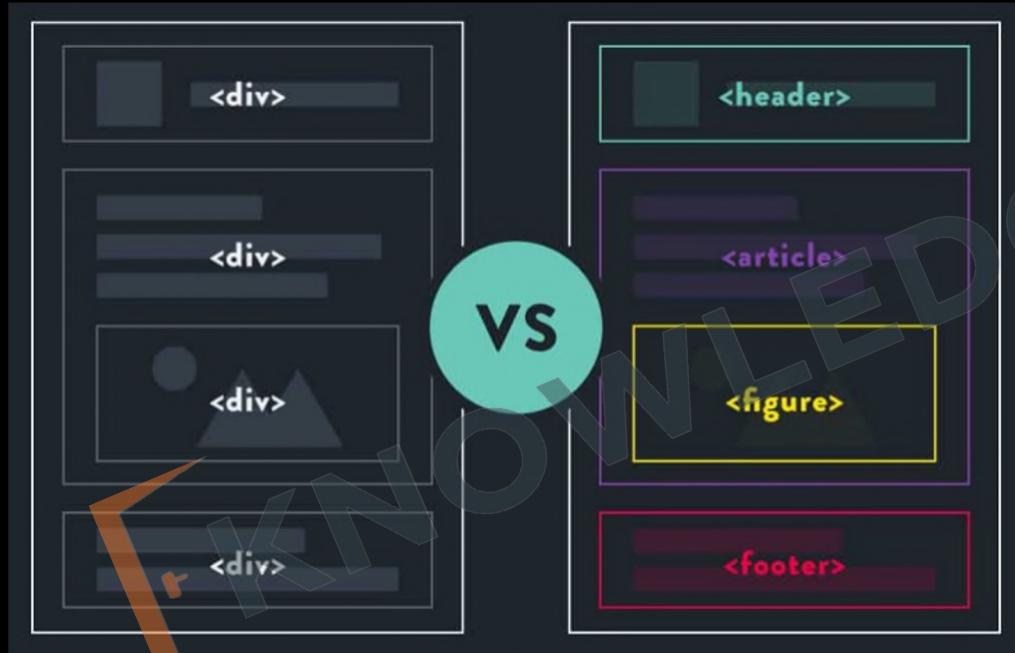
Good SEO,  
Use Relevant correct tags for specific uses

# Level 4

## HTML and Project Structure

Possible but not good

good convention



# 1. Semantic Tags

# 1.1 Semantic/Non-Semantic Tags

## Semantic Tags

- Meaningful: Describe content.
- SEO: Good for search engines.
- Accessibility: Useful for screen readers.
- Examples: `<header>`, `<footer>`, `<article>`, `<section>`, `<nav>`.

## Non-Semantic Tags

- Generic: No specific meaning.
- For Styling: Used for layout.
- No SEO: Not SEO-friendly.
- Examples: `<div>`, `<span>`, `<i>`, `<b>`.

If good Semantics, SEO

will show users not just the website name but its inside sections/navigations at search options only

# Level 4

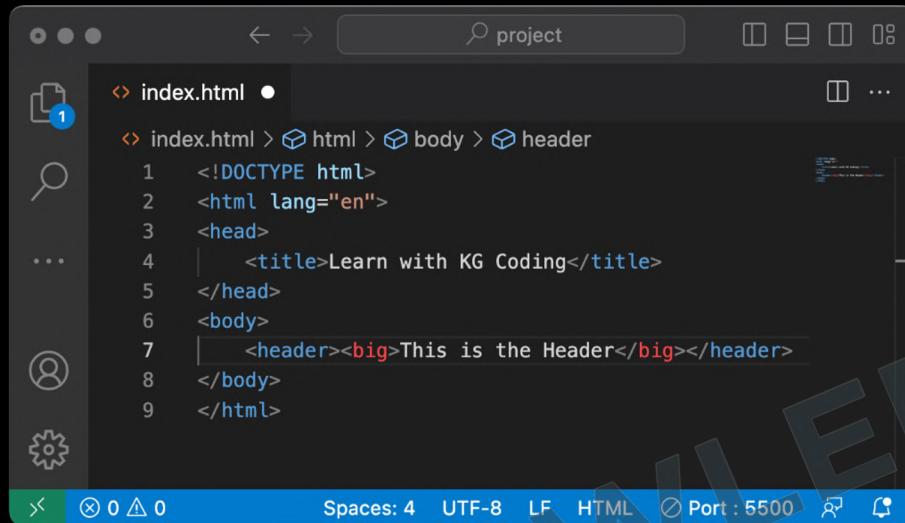
## HTML and Project Structure



ASIDE :  
not sooo important

2. Body  
Tags

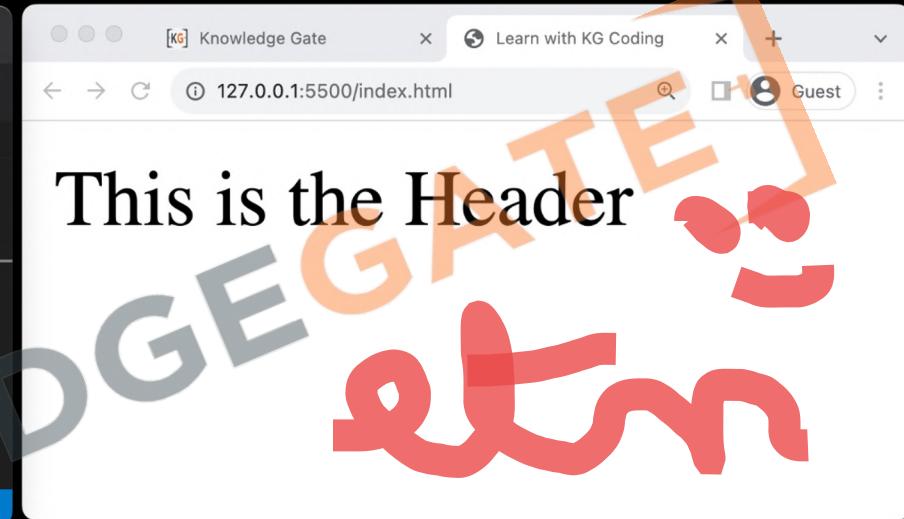
# 2.1 Header Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <header><big>This is the Header</big></header>
</body>
</html>
```

The code editor has a dark theme with light-colored text. It includes icons for file operations, search, and help. The bottom status bar shows "Spaces: 4", "UTF-8", "LF", "HTML", "Port : 5500", and a refresh icon.



1. Purpose: Used to contain introductory content or navigation links.
2. Semantic: It's a **semantic tag**, providing meaning to the enclosed content.
3. Location: Commonly found at the **top of web pages**, but can also appear within **<article>** or **<section>** tags.
4. Multiple Instances: Can be used more than once on a page within different sections.

# 2.2 Main Tag

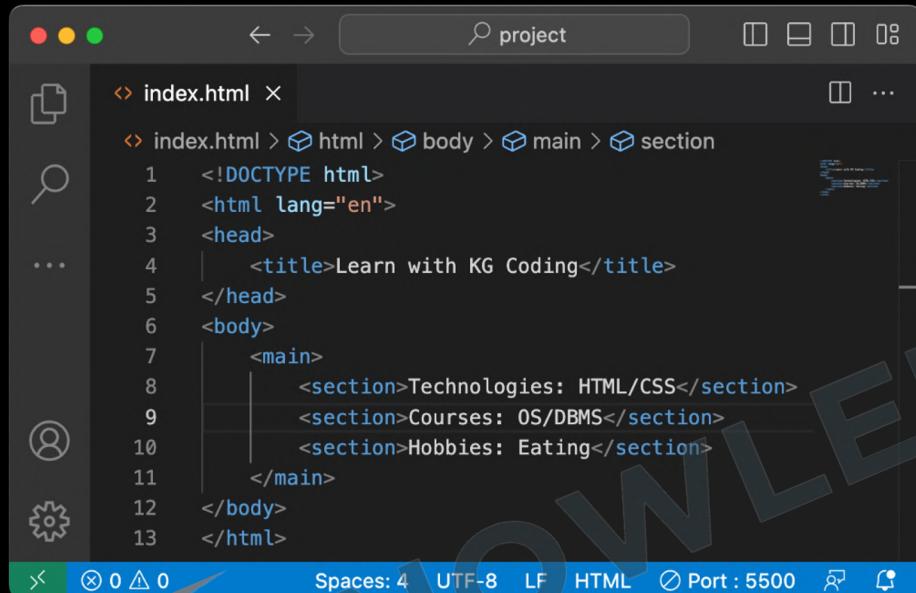
The image shows a code editor on the left and a browser window on the right. The code editor displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <header><big>This is the Header</big></header>
    <hr><main>This is main space</main>
</body>
</html>
```

The browser window shows the rendered HTML. It has a header with the text "This is the Header" and a main content area below it with the text "This is main space". A large watermark reading "KNOWLEDGE GATE" is diagonally across the center.

1. **Purpose:** Encloses the primary content of a webpage.
2. **Semantic:** Adds meaning, indicating the main content area.
3. **Unique:** Should appear only once per page.
4. **Accessibility:** Helps screen readers identify key content.
5. **Not for Sidebars:** Excludes content repeated across multiple pages like site navigation or footer.

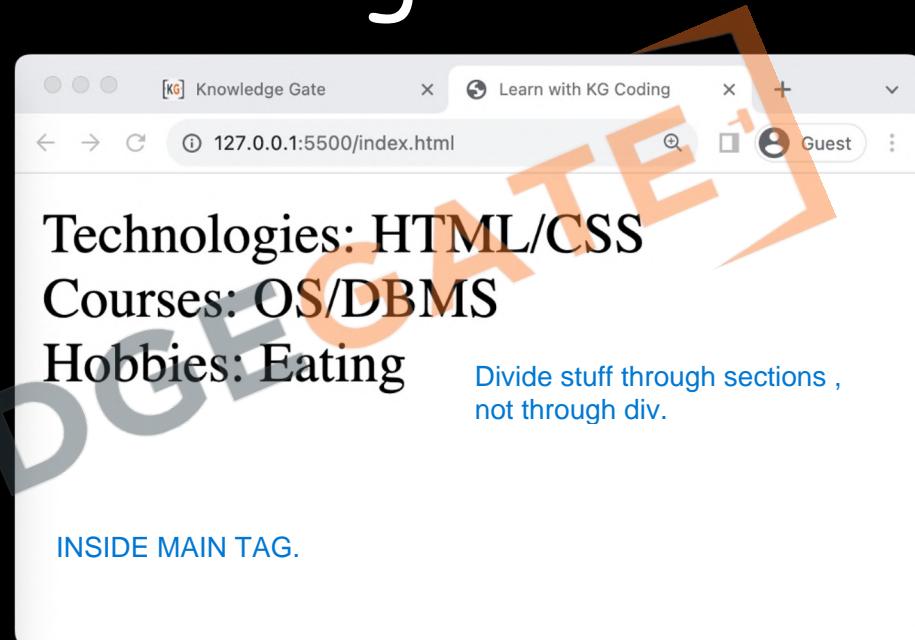
## 2.2.1 Section Tag



A screenshot of a code editor window titled "index.html". The code is as follows:

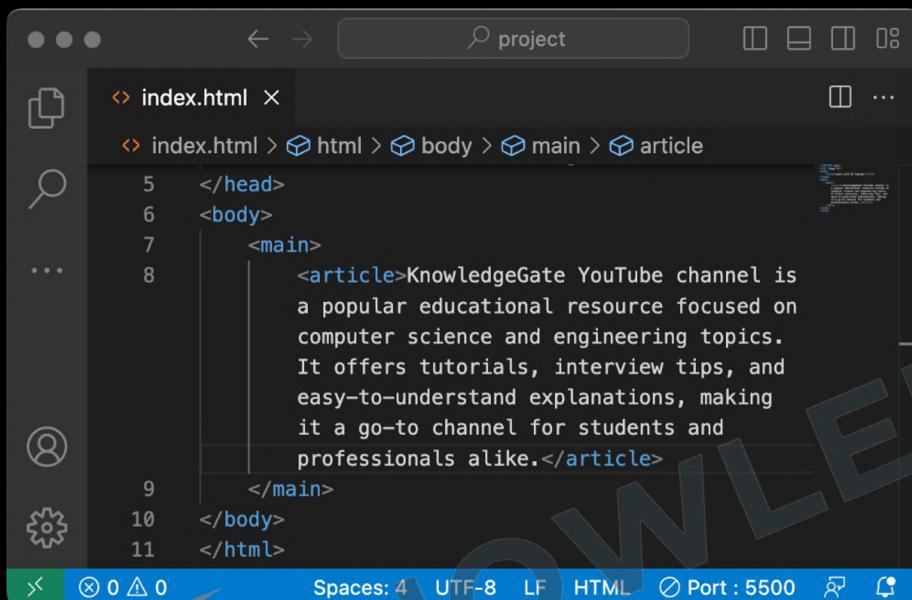
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <main>
        <section>Technologies: HTML/CSS</section>
        <section>Courses: OS/DBMS</section>
        <section>Hobbies: Eating</section>
    </main>
</body>
</html>
```

The editor has a dark theme with light-colored syntax highlighting. A sidebar on the left contains icons for file operations like new, open, save, and search.

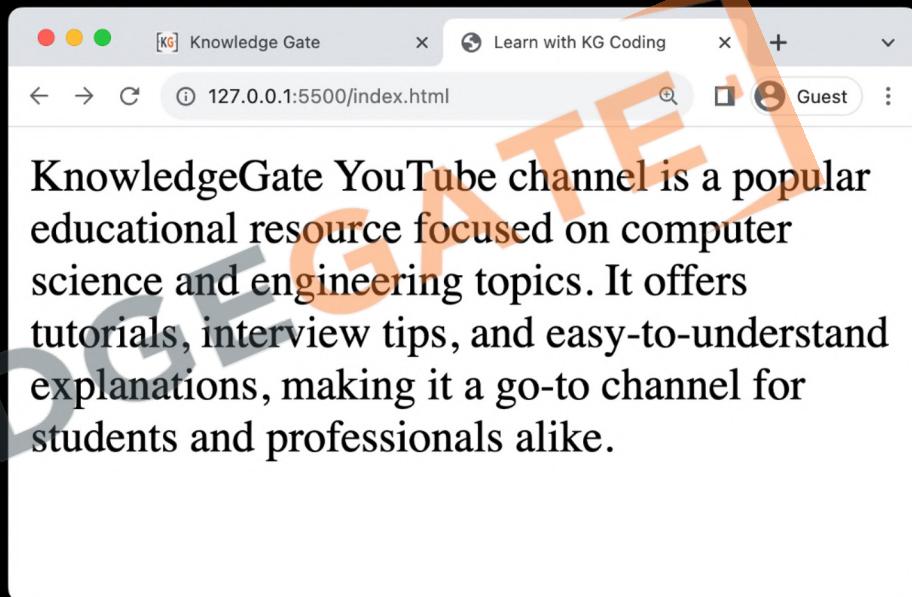


1. **Purpose:** Groups related content in a distinct section.
2. **Semantic:** Adds structure and meaning.
3. **Headers:** Often used with a heading `<h1>` to `<h6>` to indicate section topic.
4. **Nested:** Can be nested within other `<section>` or `<article>` tags.

## 2.2.2 Article Tag

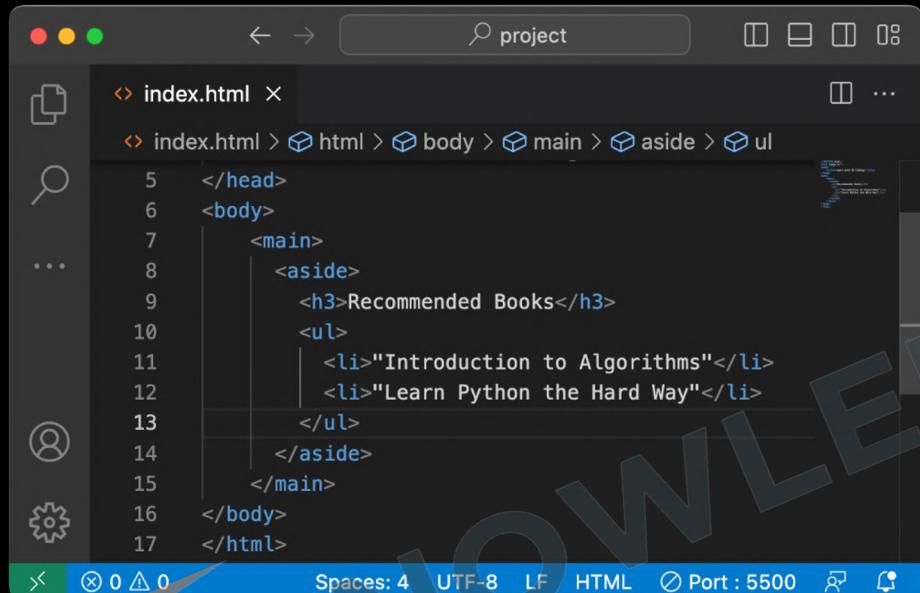


```
</head>
<body>
  <main>
    <article>KnowledgeGate YouTube channel is
    a popular educational resource focused on
    computer science and engineering topics.
    It offers tutorials, interview tips, and
    easy-to-understand explanations, making
    it a go-to channel for students and
    professionals alike.</article>
  </main>
</body>
</html>
```



1. Purpose: Encloses content that stands alone, like a blog post or news story.
2. Semantic: Provides contextual meaning.
3. Independence: Content should make sense even if taken out of the page context.
4. Multiple Instances: Can be used multiple times on the same page

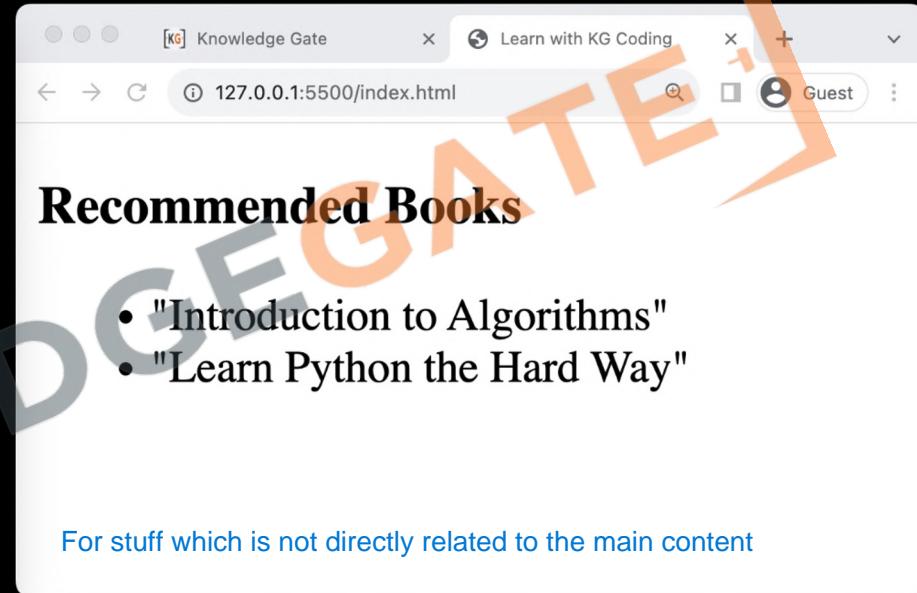
## 2.2.3 Aside Tag



A screenshot of a code editor showing the file `index.html`. The code structure is as follows:

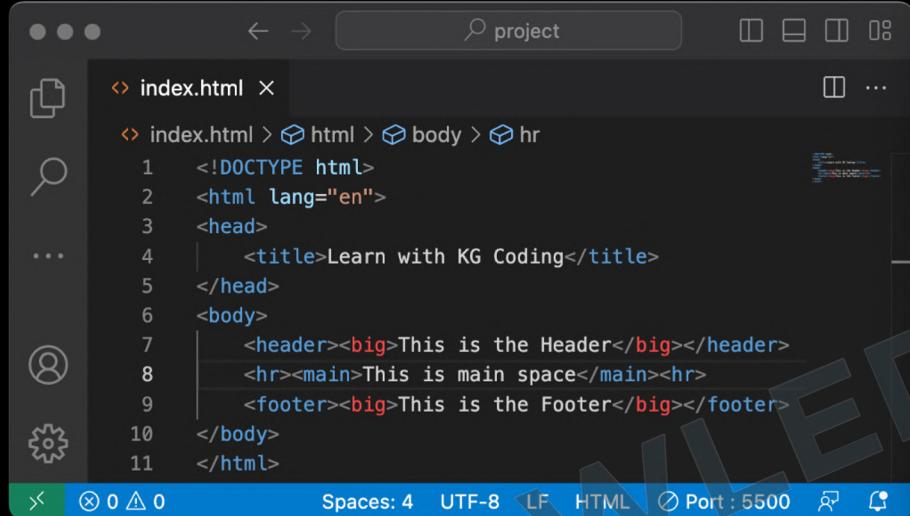
```
</head>
<body>
  <main>
    <aside>
      <h3>Recommended Books</h3>
      <ul>
        <li>"Introduction to Algorithms"</li>
        <li>"Learn Python the Hard Way"</li>
      </ul>
    </aside>
  </main>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. A sidebar on the left contains icons for file, search, and user.



1. **Purpose:** Contains sidebar or supplementary content.
2. **Semantic:** Indicates content tangentially related to the main content. tangent..just touches the circle ,not cross it
3. **Not Crucial:** Content is not essential to understanding the main content.
4. **Examples:** Could hold **widgets**, quotes, or ads.

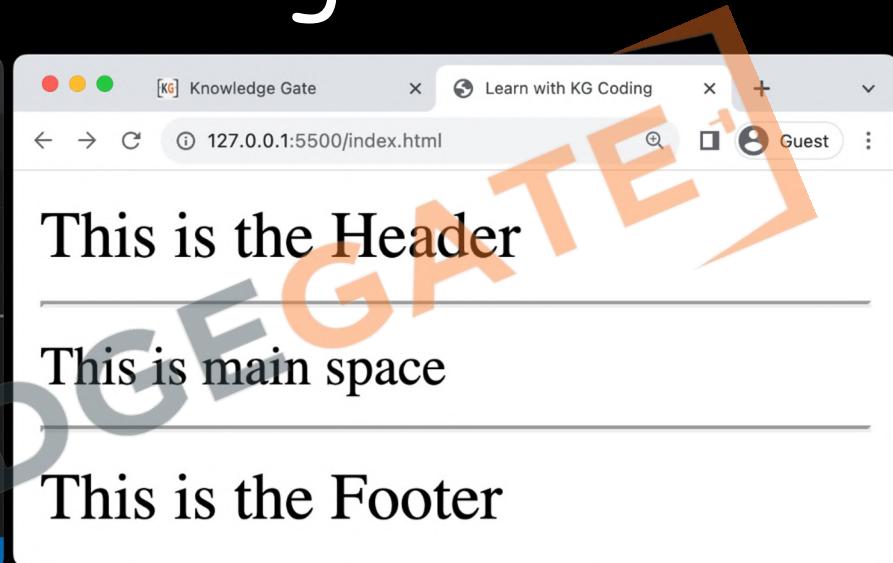
# 2.3 Footer Tag



A screenshot of a code editor showing the file `index.html`. The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <header><big>This is the Header</big></header>
    <hr><main>This is main space</main><hr>
    <footer><big>This is the Footer</big></footer>
</body>
</html>
```

The code editor interface includes a sidebar with icons for file, search, and user, and a bottom bar with tabs for Spaces: 4, UTF-8, LF, HTML, Port: 5500, and a refresh icon.

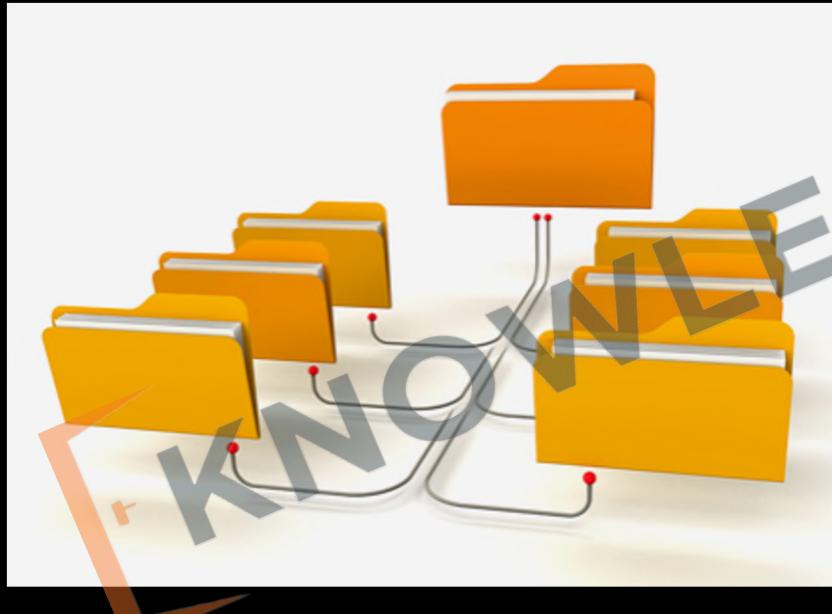


1. **Purpose:** For footer content like extra info or links.
2. **Semantic:** Provides meaning to enclosed content.
3. **Location:** Typically at the **bottom** of pages or sections.
4. **Content:** Includes copyrights, contact info, and social links.
5. **Multiple Instances:** Can be used more than once on a page.

"Foot" == Bottom

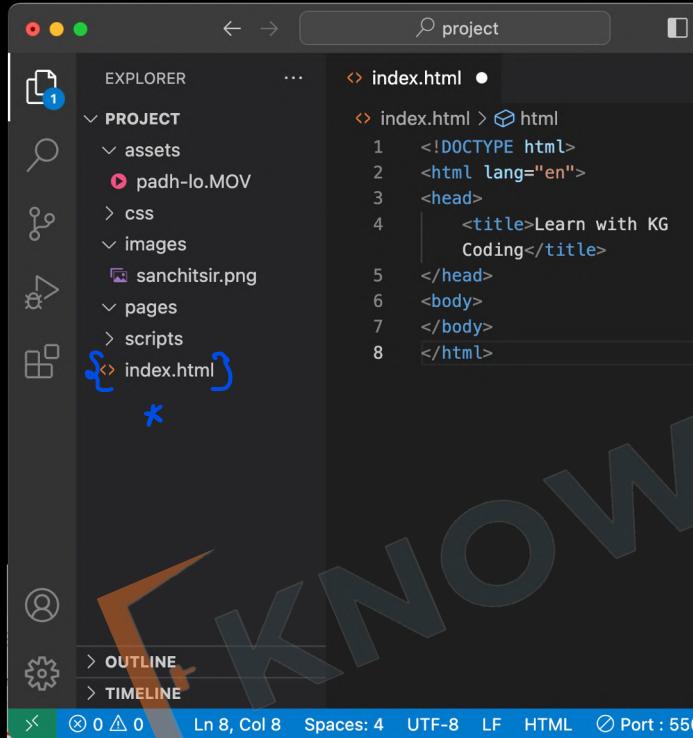
# Level 4

HTML and Project Structure



KNOWLEDGE GATE<sup>1</sup>  
**3. Folder  
Structure**

# 3.1 Recommended Folder Structure



1. **Root Directory:** Main folder containing all website files.
2. **HTML Files:** Store main .html files at the root level for easy access.
3. **CSS Folder:** Create a css/ folder for all Cascading Style Sheets.
4. **JS Folder:** Use a scripts/ folder for JavaScript files.
5. **Images Folder:** Store images in an images/ or images/ folder.
6. **Assets:** Other assets like fonts can go in an assets/ folder.
7. **Sub-directories:** For multi-page websites, use sub-folders to categorize content.

# Level 4

HTML and Project Structure



KNOWLEDGE GATE<sup>1</sup>  
4. More  
Tags

# 4.1 Navigation Tags

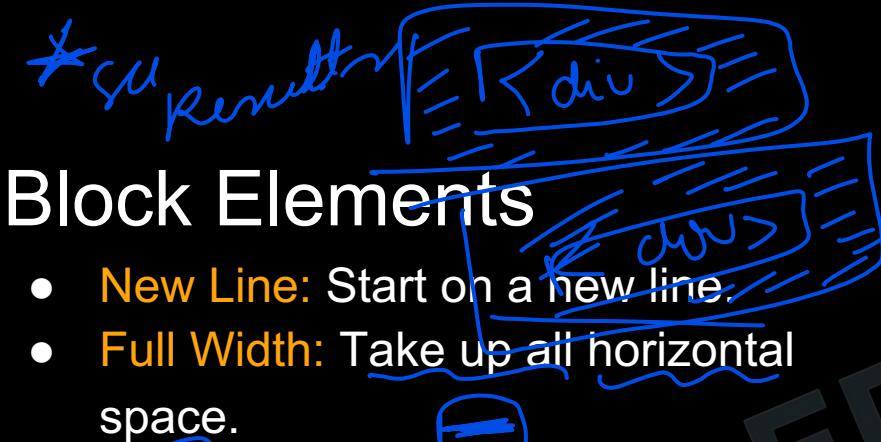
The image shows a code editor on the left and a web browser on the right. The code editor displays the file 'index.html' with the following content:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <nav>
        <a href="#home">Home</a><br>
        <a href="#about">About</a><br>
        <a href="#services">Services</a><br>
        <a href="#contact">Contact</a><br>
    </nav>
</body>
</html>
```

The browser window shows the rendered HTML with four underlined links: Home, About, Services, and Contact. A large watermark 'KNOWLEDGE GATE' is diagonally across the center. A blue checkmark is placed over the first point in the list below.

1. Purpose: Encloses navigation links or menus. ✓
2. Semantic: Signals that the content is meant for navigating the site. insides sections of the webpage, not routing.
3. Common Content: Usually contains lists `<ul>`, `<ol>` of links `<a>`.
4. Accessibility: Aids screen readers in identifying site navigation.

# 4.2 Block / Inline Elements



## Block Elements

- **New Line:** Start on a new line.
- **Full Width:** Take up all horizontal space.
- **Styling:** Can have margins and padding.
- **Size:** Width and height can be set.
- **Examples:** `<div>`, `<p>`, `<h1>`, `<ul>`, `<li>`.



## Inline Elements

- **Flow:** Stay in line with text.
- **Width:** Just as wide as the content.
- **No Break:** No new line between elements.
- **Limited Styling:** Can't set size easily.
- **Examples:** `<span>`, `<a>`, `<strong>`, `<em>`, `<img>`.

# \*play with input 4.3 Div Tags

A screenshot of a code editor window titled "index.html". The code is as follows:

```
<head>
  <title>Learn with KG Coding</title>
</head>
<body>
  <div>
    <p> Lorem ipsum dolor sit amet
    consectetur adipisicing elit. Itaque quae
    veritatis, repellendus nam adipisci fuga
    nulla eos nobis.</p>
  </div>
</body>
</html>
```

The word "block" is handwritten in blue across the bottom of the code editor.

A screenshot of a browser window titled "Knowledge Gate" showing the URL "127.0.0.1:5500/index.html". The content is:

Learn with KG Coding

LOREM IPSUM DOLOR SIT AMET CONSECTETUR  
ADIPISICING ELIT. ITAQUE QUAE VERITATIS,  
REPELLENDUS NAM ADIPISCI FUGA NULLA EOS NOBIS.

Handwritten notes in blue:

- "logical parking" is written above the first paragraph.
- "needed" is written above the second paragraph.
- "possible" is written below the second paragraph.

1. Purpose: Acts as a container for other **HTML** elements. *CSS transition pos. ch. general + day.*
2. Non-Semantic: Doesn't provide inherent meaning to enclosed content. *general + day.*
3. Styling: Commonly used for layout and styling via **CSS**.
4. Flexibility: Highly versatile and can be customized using classes or IDs.

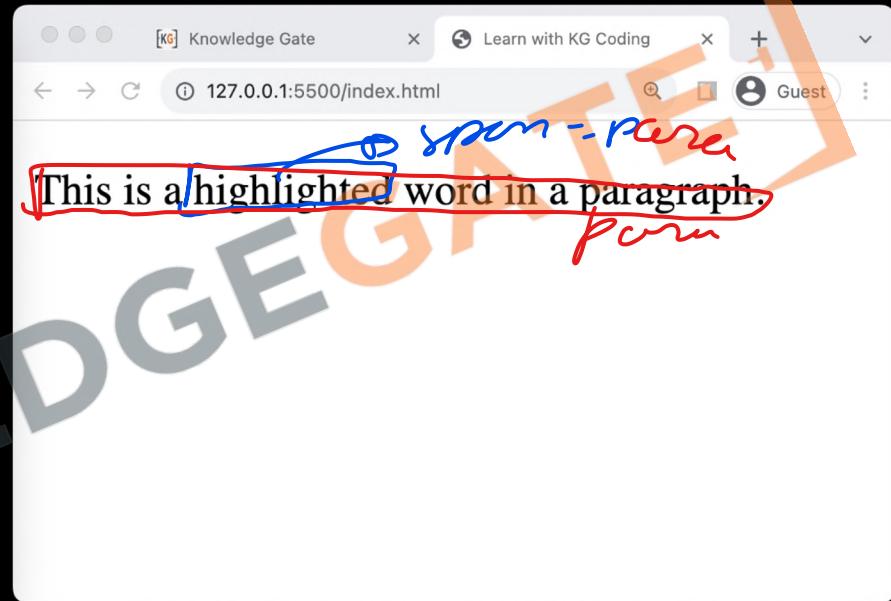
# 4.4 Span Tags

\*div with inline:)

A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <p>This is a <span class="highlight">highlighted</span> word in a paragraph.</p>
</body>
</html>
```

The word "highlighted" is highlighted with a blue background and white text. The status bar at the bottom shows "Spaces: 4", "UTF-8", "LF", "HTML", "Port : 5500", and icons for save, undo, and redo.



1. Purpose: Used for inline elements to style or manipulate a portion of text.
2. Non-Semantic: Doesn't add specific meaning to the enclosed text.
3. Styling: Commonly used for changing color, font, or adding effects via CSS.
4. Inline Nature: Doesn't break text flow or create a new block-level element.

# Level 4 Revision

## HTML and Project Structure

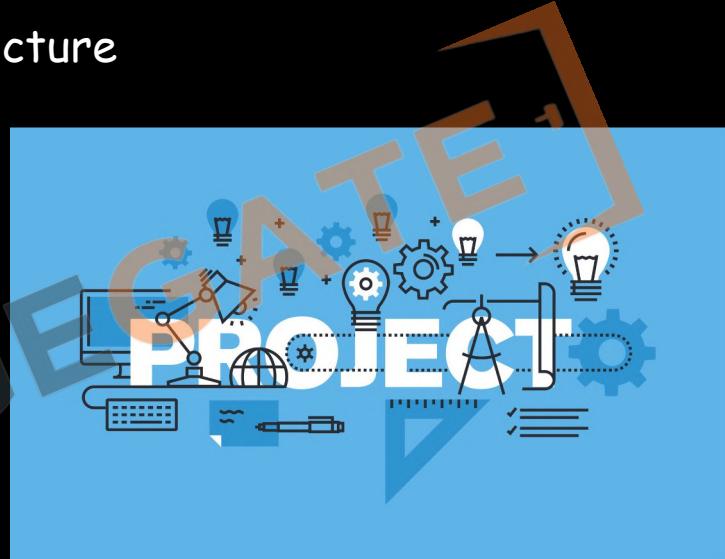
1. Semantic Tags
  1. Semantic / Non-Semantic Tags
2. Body Tags
  1. Header Tag
  2. Main Tag
    1. Section Tag
    2. Article Tag
    3. Aside Tag
  3. Footer Tag
3. Folder Structure
  1. Recommended Folder structure
4. More Tags
  1. Navigation tags
  2. Block / Inline Elements
  3. Div tags
  4. Span Tags



# Project Level 4

## HTML and Project Structure

1. Create a **page** with header, footer, main(section, article, aside tag).
2. Make sure the project from level 3 has correct **folder** structure.
3. Create **groupings** of multiple tags using div.
4. Create **navigation** to important sections of your page.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Level 5

## List, Tables & Forms

1. List Tag
  1. Ordered Lists
  2. Types of Ordered Lists
  3. Unordered Lists
2. Table Tag
  1. `<tr>`, `<td>`, `<th>` tags
  2. Captions
  3. Col spans
3. Forms
  1. Input tag
  2. Action Attributes
  3. Name and Value Property
  4. Label Tag
  5. Exploring Types
4. iFrame Tag
  1. Using iFrames

Don't Ratto.  
IDE will tell list ways etc.  
or google /it

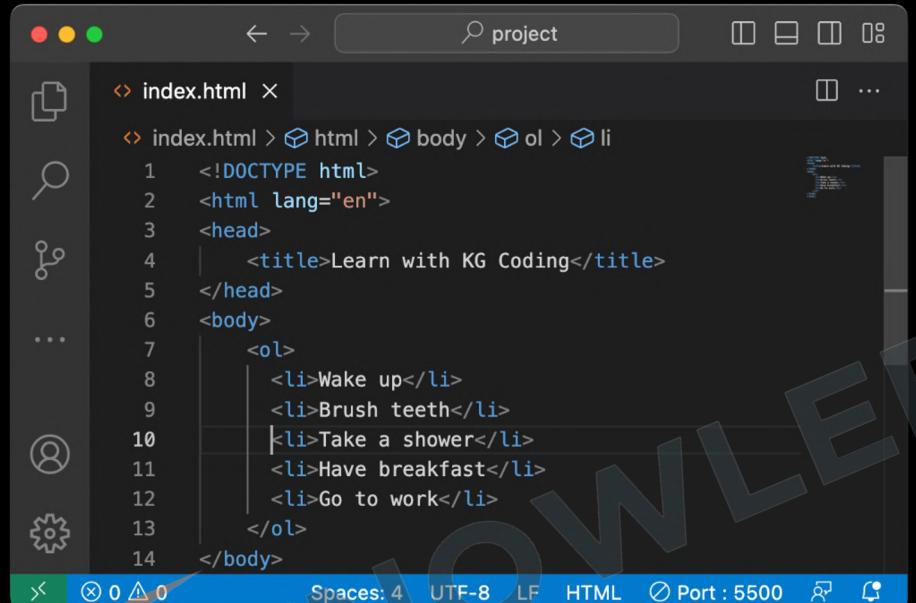
# Level 5

## List, Tables & Forms



1. List Tag

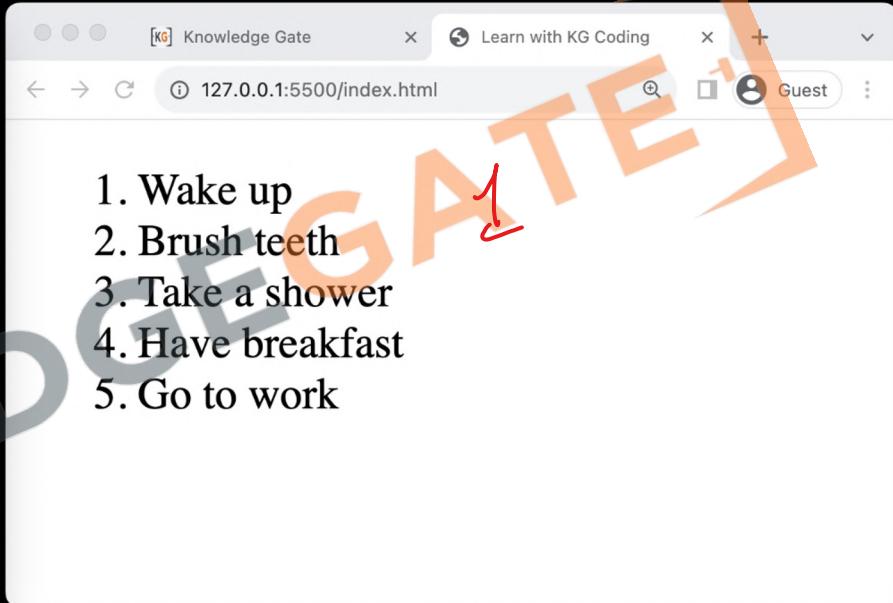
# 1.1 Ordered Lists



A screenshot of a code editor showing the file `index.html`. The code contains an ordered list:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <ol>
        <li>Wake up</li>
        <li>Brush teeth</li>
        <li>Take a shower</li>
        <li>Have breakfast</li>
        <li>Go to work</li>
    </ol>
</body>
```

The status bar at the bottom shows "Spaces: 4" and "Port: 5500".



1. Purpose: Used for creating lists with items that have a specific order.
2. Default: Items are automatically numbered.
3. Nesting: Can be nested within other lists.

# 1.2 Types of Ordered Lists

## Ordered Lists

- **Numeric:** Default type, (1, 2, 3, ...)  
Attribute: `type="1"`
- **Uppercase Letters:** (A, B, C, ...)  
Attribute: `type="A"`
- **Lowercase Letters:** (a, b, c, ...)  
Attribute: `type="a"`
- **Uppercase Roman:** (I, II, III, ...)  
Attribute: `type="I"`
- **Lowercase Roman:** (i, ii, iii, ...)  
Attribute: `type="i"`

A. Apple  
B. Banana  
C. Cherry  
D. Dragonfruit

a. Apple  
b. Banana  
c. Cherry  
d. Dragonfruit

I. Apple  
II. Banana  
III. Cherry  
IV. Dragonfruit

i. Apple  
ii. Banana  
iii. Cherry  
iv. Dragonfruit

# 1.3 Unordered Lists

The image shows a code editor on the left and a browser window on the right. The code editor displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <ul>
        <li>Apple</li>
        <li>Banana</li>
        <li>Cherry</li>
        <li>Dragonfruit</li>
    </ul>
</body>
</html>
```

The browser window shows the rendered output: a bulleted list of four items: Apple, Banana, Cherry, and Dragonfruit. A large orange watermark "KNOWLEDGE GATE" is diagonally across the center. Red annotations on the right side of the browser window point to the bullet points and the list items, with the word "marked" written next to them.

1. Purpose: Used for lists where the order of items doesn't matter.
2. Default: Items are usually bulleted.
3. Nesting: Can be nested within other lists.

# Level 5

List, Tables & Forms



NON-EDGE GATE 1  
Table Tag

# 2.1 <tr>, <td>, <th> Tags

The image shows a code editor on the left and a browser window on the right. The code editor displays the HTML file 'index.html' with the following content:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Learn with KG Coding</title>
  </head>
  <body>
    <table border="1">
      <tr>
        <th>Name</th>
        <th>Age</th>
        <th>Email</th>
      </tr>
      <tr>
        <td>John</td>
        <td>30</td>
        <td>john@email.com</td>
      </tr>
    </table>
  </body>
</html>
```

The browser window shows the rendered table with the following data:

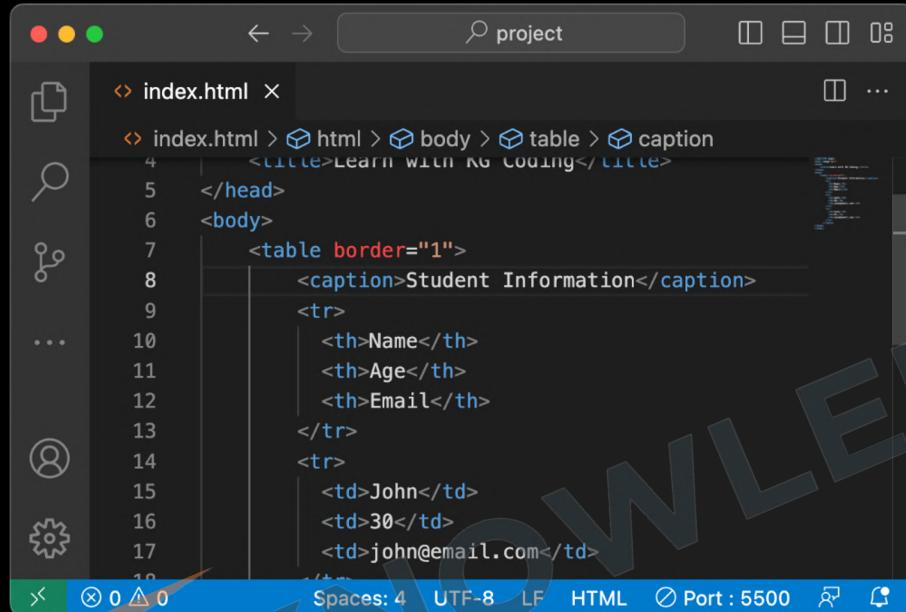
Name	Age	Email
John	30	john@email.com
Jane	25	jane@email.com

Handwritten annotations in the browser window include:

- A yellow circle around the word 'Table' in the code editor, with a yellow arrow pointing to the word 'Table' in the browser.
- A blue circle around the word 'border' in the code editor, with a blue arrow pointing to the 'border="1"' attribute in the browser.
- A red circle around the word 'td' in the code editor, with a red arrow pointing to the 'td' cells in the browser.
- A purple circle around the word 'th' in the code editor, with a purple arrow pointing to the 'th' header cells in the browser.

1. **<tr> Table Row** : Used to define a row in an HTML table.
2. **<th> Table Header** : Used for header cells within a row.  
Text is bold and centered by default.
3. **<td> Table Data** : This Holds the actual data.

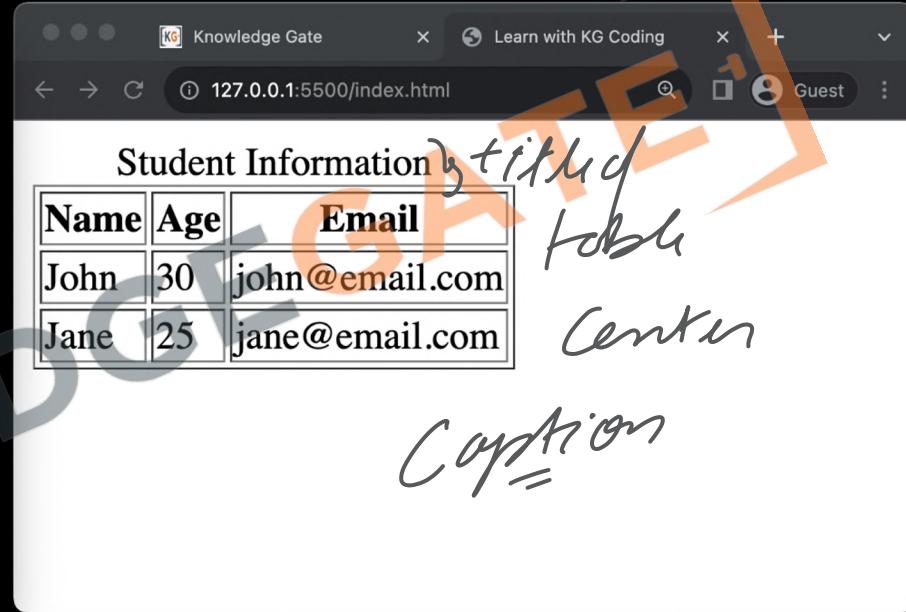
## 2.2 Captions



The screenshot shows a code editor window with the file "index.html" open. The code defines a table with a caption:

```
<head><title>Learn HTML Using Coding</title>
</head>
<body>
    <table border="1">
        <caption>Student Information</caption>
        <tr>
            <th>Name</th>
            <th>Age</th>
            <th>Email</th>
        </tr>
        <tr>
            <td>John</td>
            <td>30</td>
            <td>john@email.com</td>
        </tr>
    </table>
</body>
```

The status bar at the bottom indicates "Spaces: 4" and "Port: 5500".

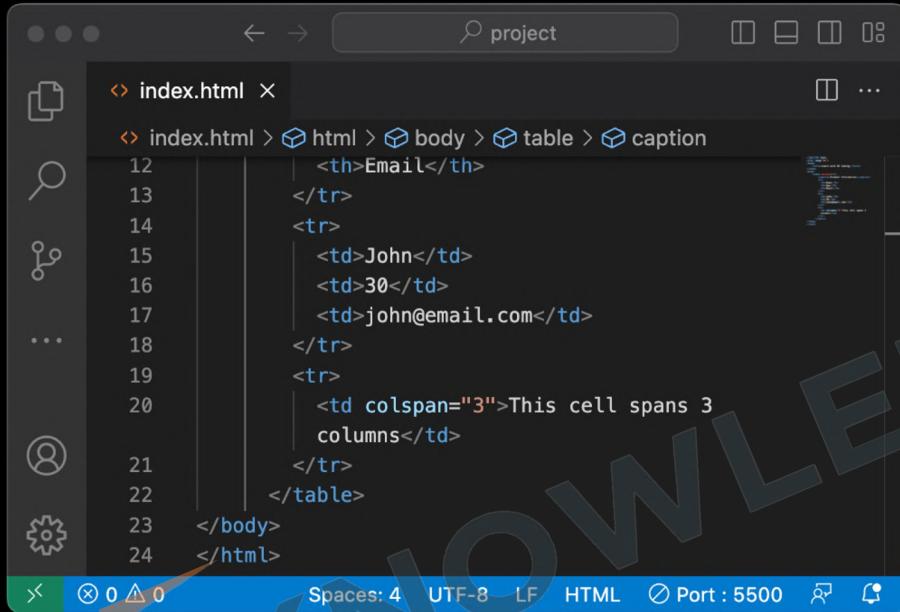


The screenshot shows a browser window displaying the "index.html" page. The table has a caption "Student Information". Handwritten annotations include "Caption" under the caption, "Center" next to the table, and "table" next to the caption.

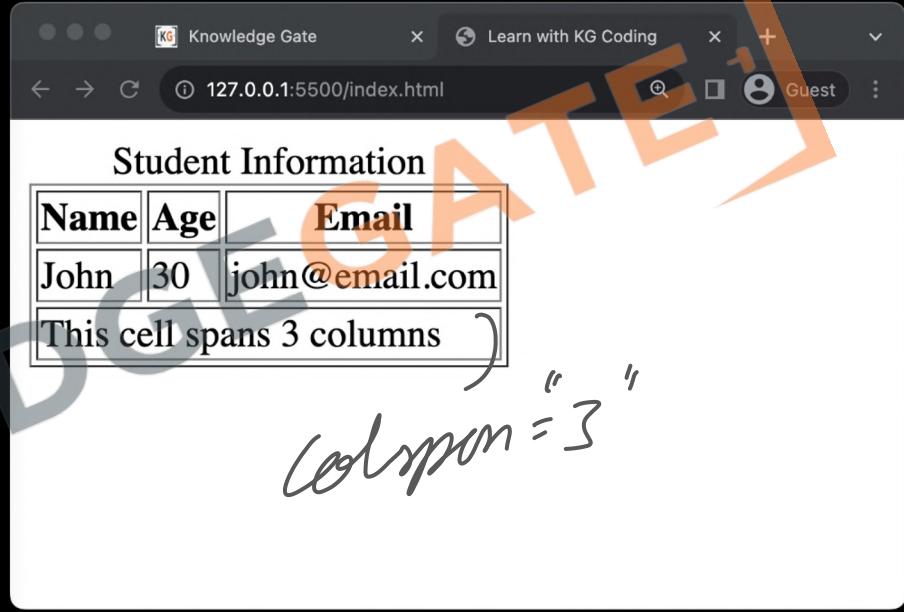
Name	Age	Email
John	30	john@email.com
Jane	25	jane@email.com

1. **Purpose:** Provides a title or description for a table.
2. **Placement:** Must be inserted **immediately after** the `<table>` opening tag.
3. **Alignment:** **Centered** above the table by default.
4. **Accessibility:** Helps screen readers understand the table's purpose.

# 2.3 Col Spans



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html>
<html>
    <head>
        <title>Student Information</title>
    </head>
    <body>
        <table border="1">
            <caption>Student Information</caption>
            <thead>
                <tr>
                    <th>Name</th>
                    <th>Age</th>
                    <th>Email</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>John</td>
                    <td>30</td>
                    <td>john@email.com</td>
                </tr>
                <tr>
                    <td colspan="3">This cell spans 3 columns</td>
                </tr>
            </tbody>
        </table>
    </body>
</html>
```



Student Information

Name	Age	Email
John	30	john@email.com
This cell spans 3 columns		

colspan = 3

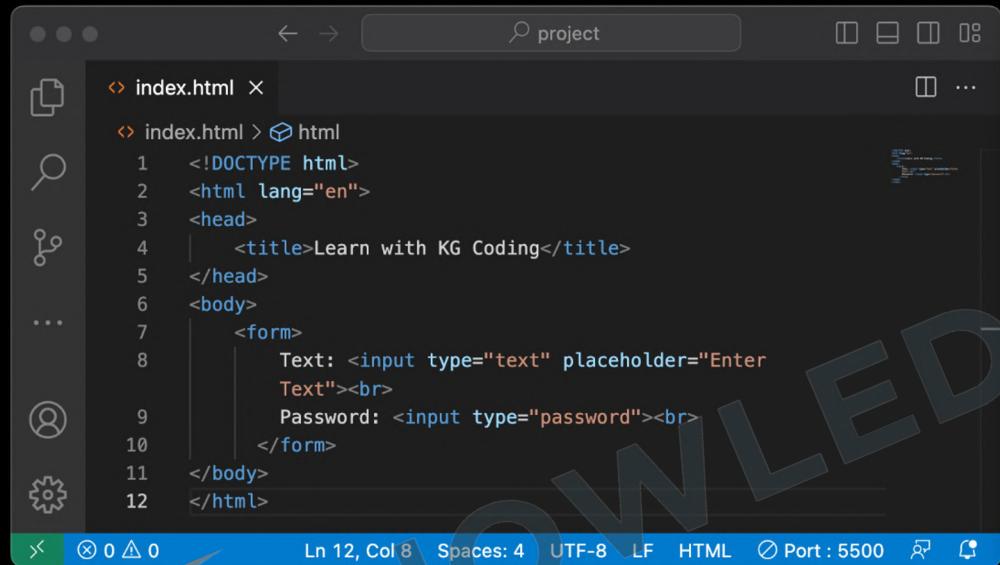
1. **Attribute:** Uses the `colspan` attribute in `<td>` or `<th>` tags.
2. **Purpose:** Allows a cell to **span multiple columns** horizontally.
3. **Alignment:** Takes the space of the **specified number of columns**.
4. **Layout:** Useful for combining cells to create complex table layouts.

# Level 5

List, Tables & Forms



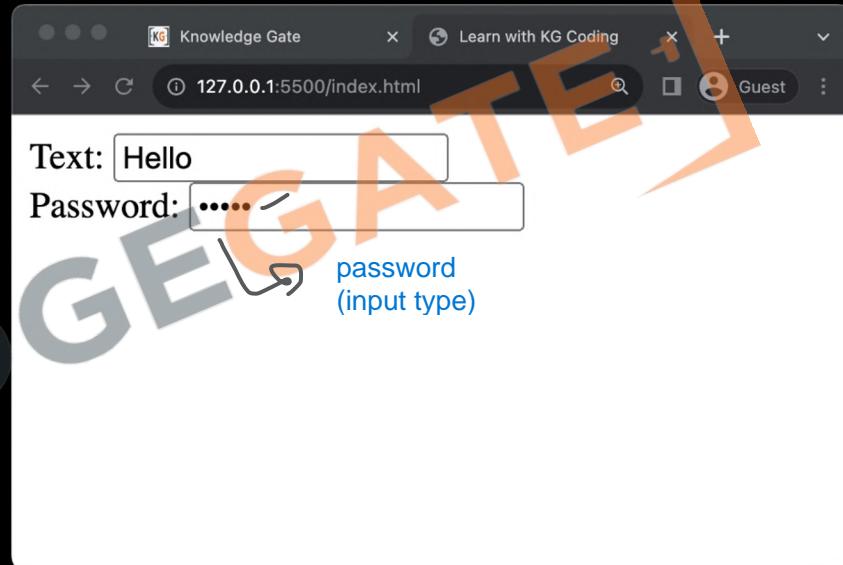
# 3.1 Input Tag



A screenshot of a code editor window titled "project". The file "index.html" is open, displaying the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
    </form>
</body>
</html>
```

The code editor interface includes a sidebar with icons for file, search, and settings, and a bottom status bar showing line 12, column 8, and other file details.



A screenshot of a web browser window titled "Knowledge Gate" showing the URL "127.0.0.1:5500/index.html". The page displays a form with two input fields. The first field is a text input containing "Hello". The second field is a password input containing ".....". A callout arrow points from the text "password (input type)" to the second input field. The browser interface includes a sidebar with icons and a bottom status bar.

1. **Purpose:** Used within a `<form>` element to collect user input.
2. **Self-Closing:** The `<input>` tag is self-closing; doesn't require a closing tag.
3. **Attributes:** Common attributes are `name`, `value`, `placeholder`, and `required`.

# 3.2 Action attribute

The image shows a code editor on the left and a browser window on the right. The code editor displays the file 'index.html' with the following content:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form action="/submit.php" method="post">
        <input type="text" name="name">
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

The browser window shows the rendered HTML with a text input field and a submit button labeled 'Submit'. A large, semi-transparent watermark reading 'KNOWLEDGE GATE' diagonally across the center of the screen.

1. **Purpose:** Specifies the URL to which the form data should be sent when submitted.
2. **Default:** If not specified, the form will be submitted to the current page's URL.
3. **Server-Side:** Usually points to a server-side script (like PHP, Python, etc.) that processes the form data.

# 3.3 Name and Value property

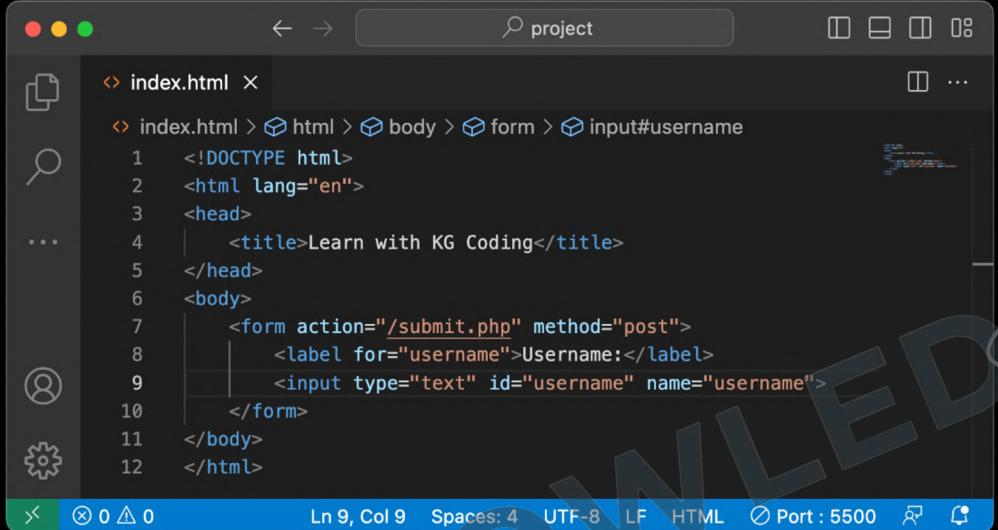
The image shows a code editor on the left and a browser window on the right. The code editor displays the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form action="/submit.php" method="post">
        <input type="text" name="username" value="John">
    </form>
</body>
</html>
```

The browser window shows a single input field containing the text "John". A large watermark reading "KNOWLEDGE GATE" diagonally across the center of the screen.

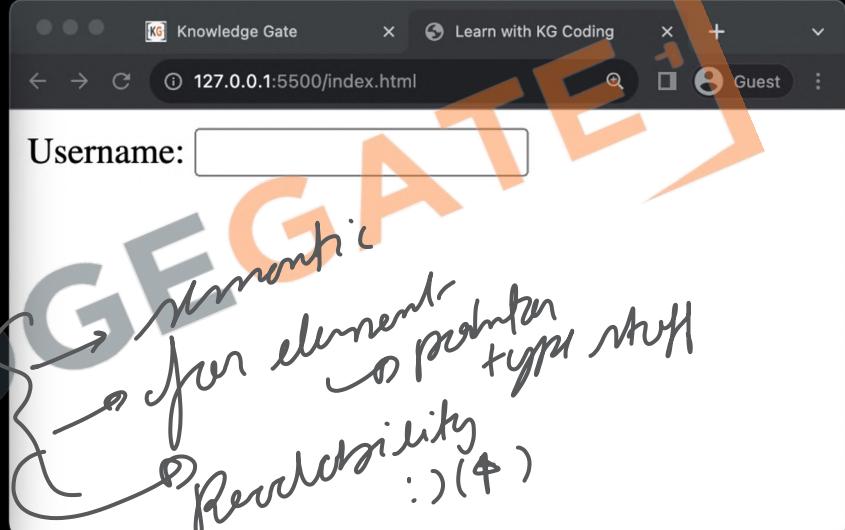
- `name` Property:
  - **ID for Data:** Identifies form elements when submitting.
  - **Unique:** Should be unique to each element for clarity.
- `value` Property:
  - **Default Data:** Sets initial value for input elements.
  - **Sent to Server:** This is the data sent when form is submitted.

# 3.4 Label Tag



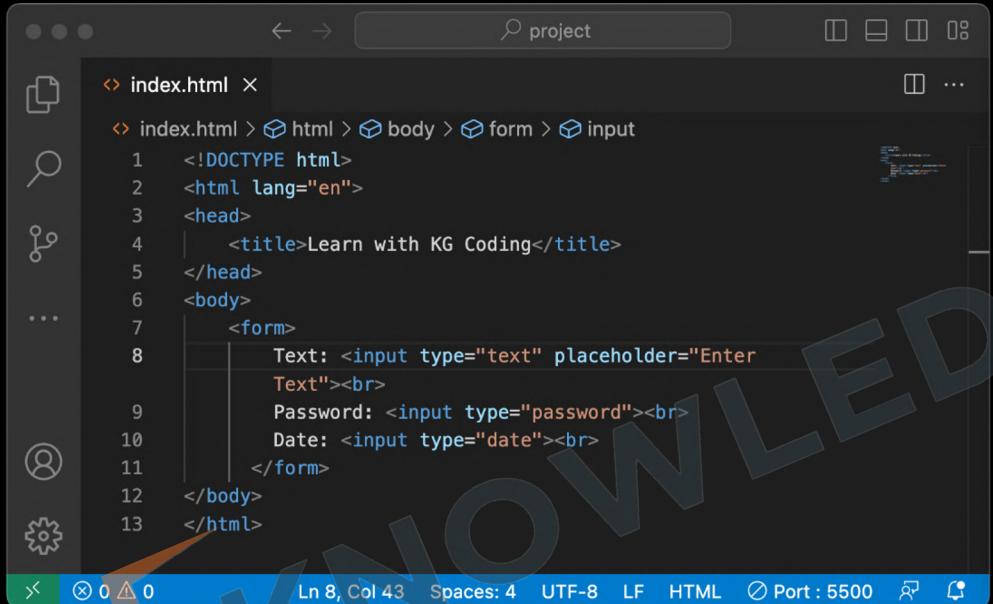
A screenshot of a code editor showing the file `index.html`. The code defines a form with a text input field and a label element:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form action="/submit.php" method="post">
        <label for="username">Username:</label>
        <input type="text" id="username" name="username">
    </form>
</body>
</html>
```



- **Purpose:** Adds a text description to form elements.
- **for Attribute:** Connects the label to a specific **form element** using the element's id.
- **Accessibility:** Makes the form more accessible.
- **Readability:** Enhances form readability and usability.

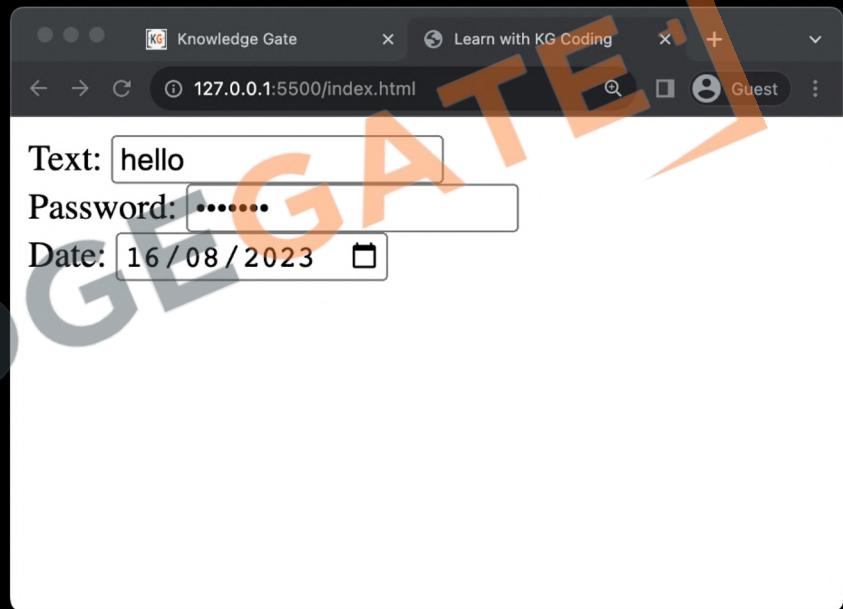
# 3.5 Input type: Date



A screenshot of a code editor window titled "index.html". The code editor shows the following HTML structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
        Date: <input type="date"><br>
    </form>
</body>
</html>
```

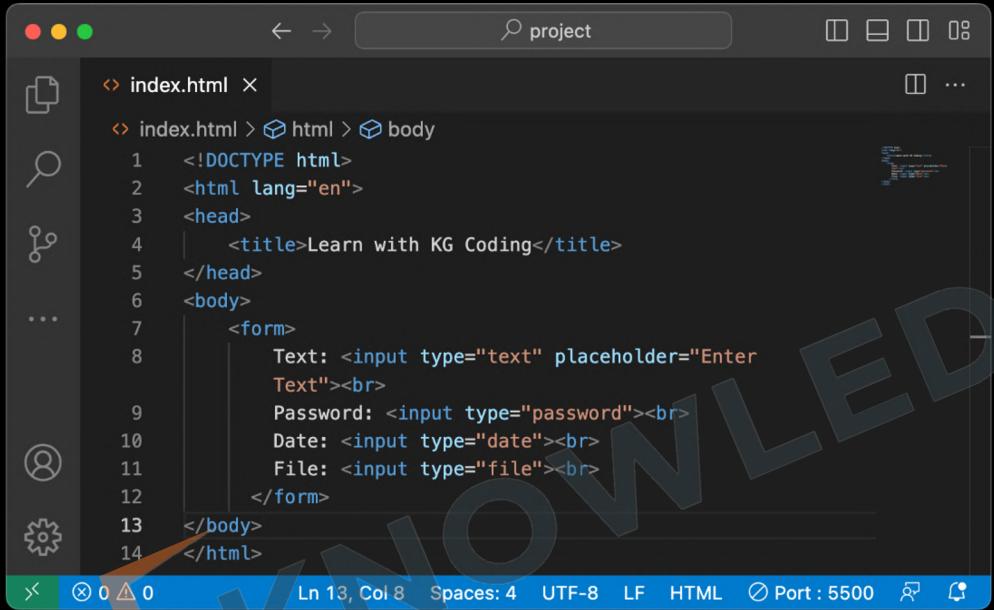
The status bar at the bottom indicates: Ln 8, Col 43, Spaces: 4, UTF-8, LF, HTML, Port : 5500.



A screenshot of a web browser window titled "Knowledge Gate". The URL is 127.0.0.1:5500/index.html. The page displays a form with three input fields:

- Text:
- Password:
- Date:

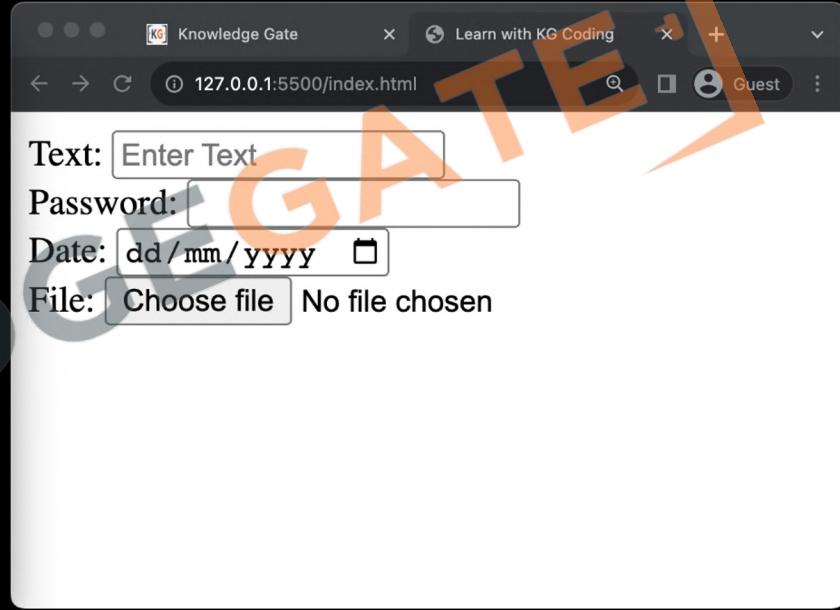
# 3.5 Input type: File



A screenshot of a code editor window titled "project". The file "index.html" is open, showing the following code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
        Date: <input type="date"><br>
        File: <input type="file"><br>
    </form>
</body>
</html>
```

The code editor has a dark theme with light-colored syntax highlighting. A sidebar on the left contains icons for file operations like new, open, save, and search.



A screenshot of a web browser window titled "Knowledge Gate" with the URL "127.0.0.1:5500/index.html". The page displays a form with four input fields:

Text:

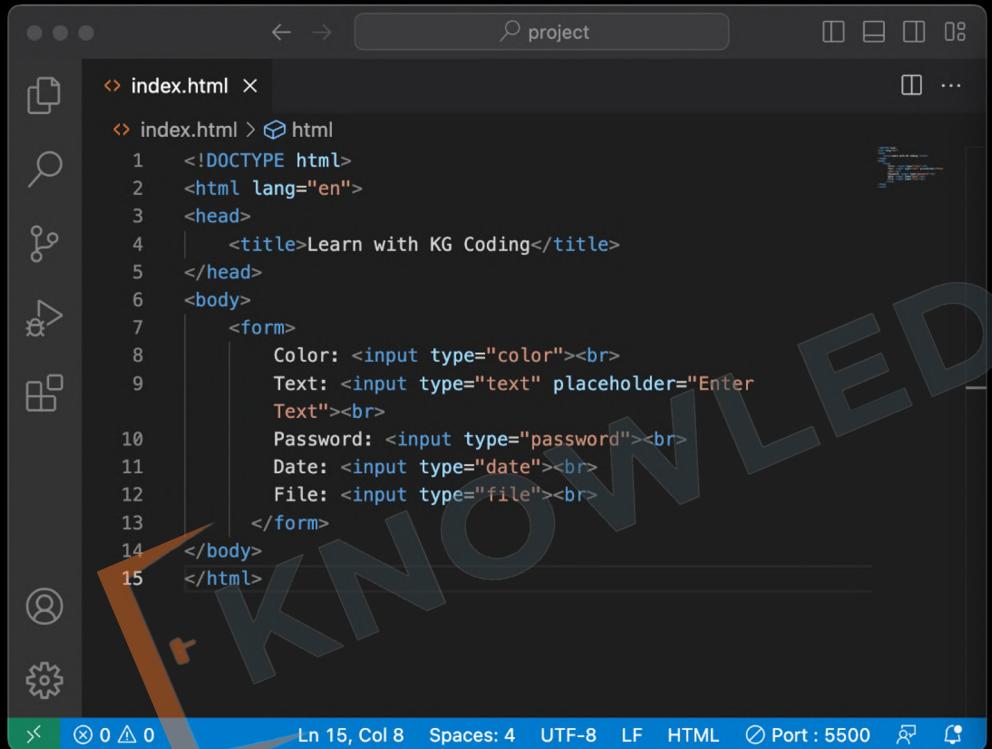
Password:

Date:

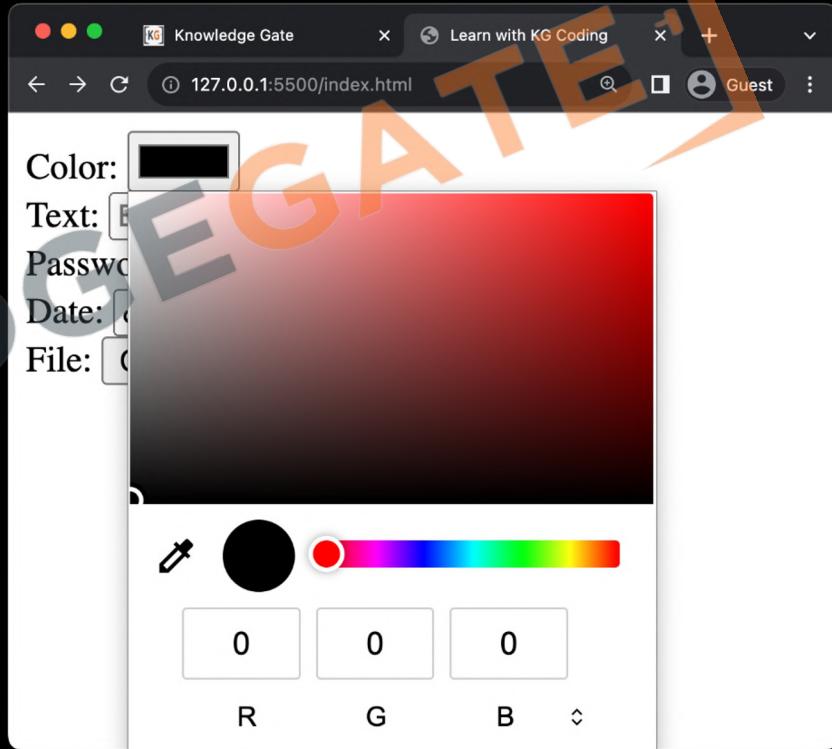
File:  Choose file No file chosen

The browser interface includes standard navigation buttons (back, forward, search) and a user profile icon.

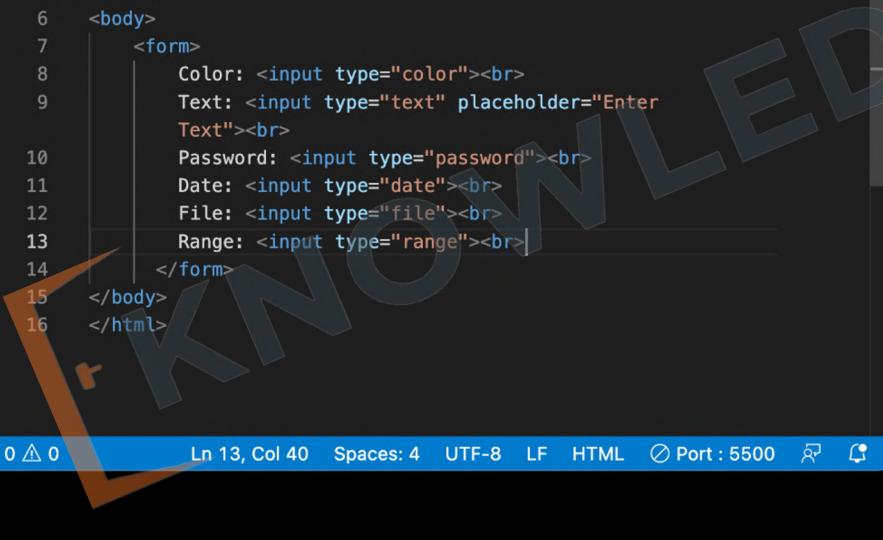
# 3.5 Input type: Color



```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Color: <input type="color"><br>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
        Date: <input type="date"><br>
        File: <input type="file"><br>
    </form>
</body>
</html>
```

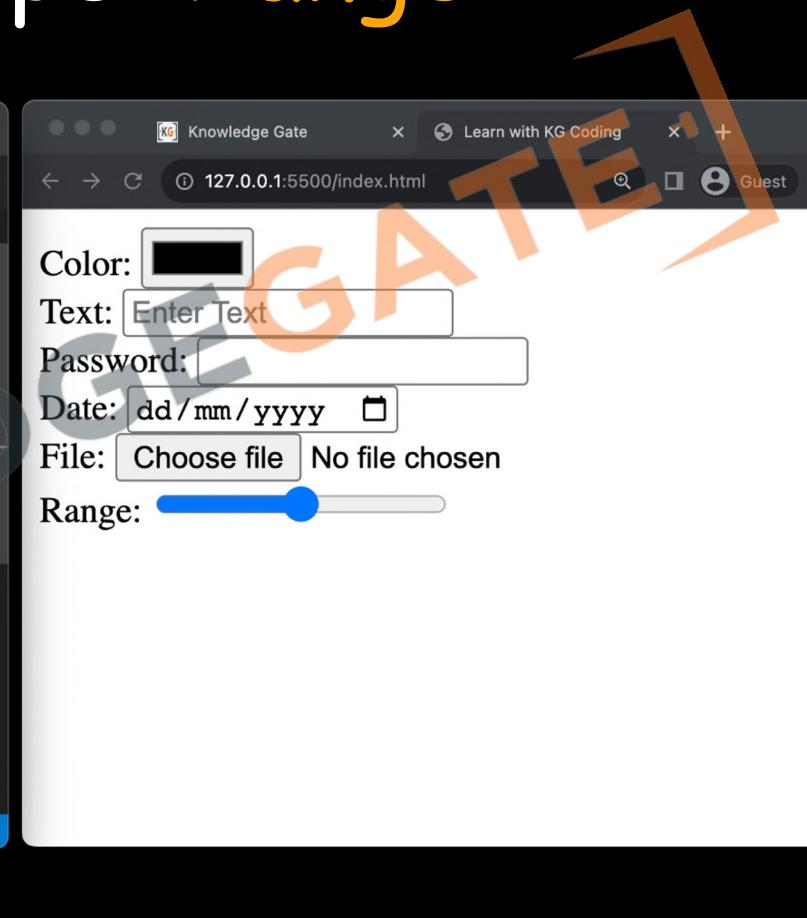


# 3.5 Input type: Range



```
project
index.html ×
index.html > html > body > form > br
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>Learn with KG Coding</title>
5  </head>
6  <body>
7    <form>
8      Color: <input type="color"><br>
9      Text: <input type="text" placeholder="Enter Text"><br>
10     Password: <input type="password"><br>
11     Date: <input type="date"><br>
12     File: <input type="file"><br>
13     Range: <input type="range"><br>
14   </form>
15 </body>
16 </html>
```

Ln 13, Col 40 Spaces: 4 UTF-8 LF HTML ⚙️ Port : 5500 🔍



Knowledge Gate x Learn with KG Coding Guest :

Color:

Text:

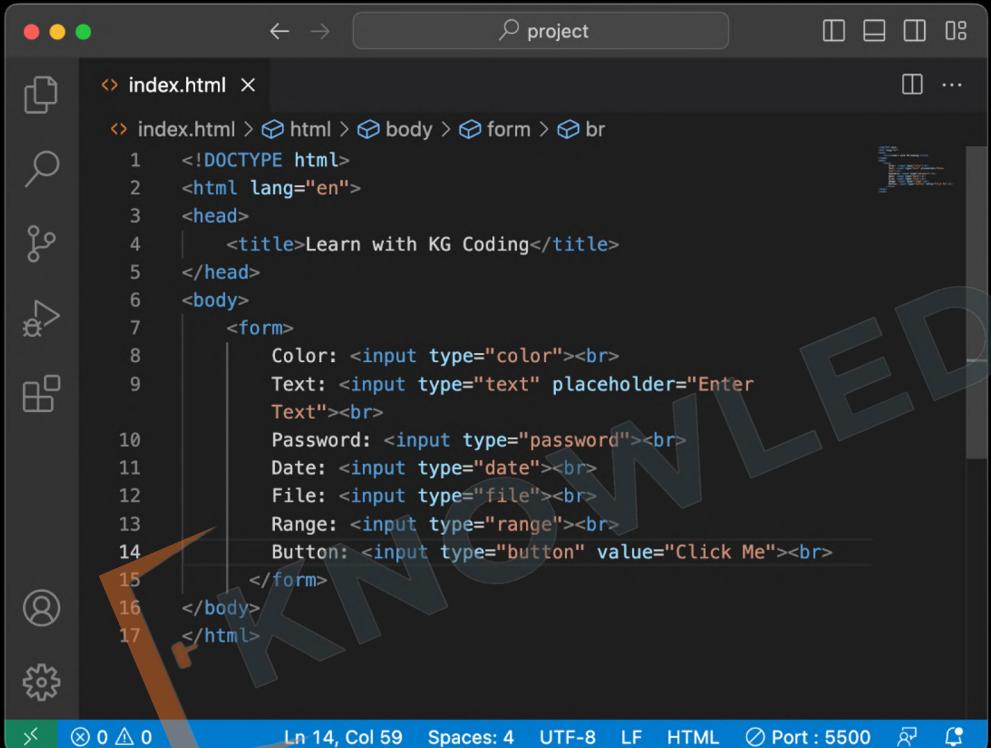
Password:

Date:

File:  Choose file No file chosen

Range:

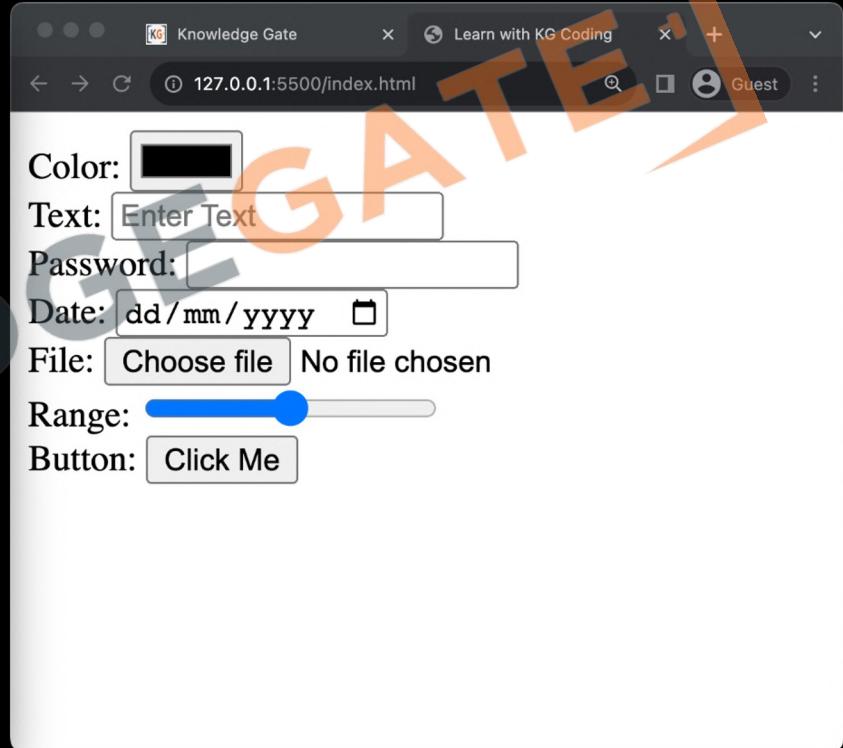
# 3.5 Input type: Button



A screenshot of a code editor showing the file `index.html`. The code defines a form with several input fields:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Color: <input type="color"><br>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
        Date: <input type="date"><br>
        File: <input type="file"><br>
        Range: <input type="range"><br>
        Button: <input type="button" value="Click Me"><br>
    </form>
</body>
</html>
```

The code editor interface includes a sidebar with various icons and a status bar at the bottom.

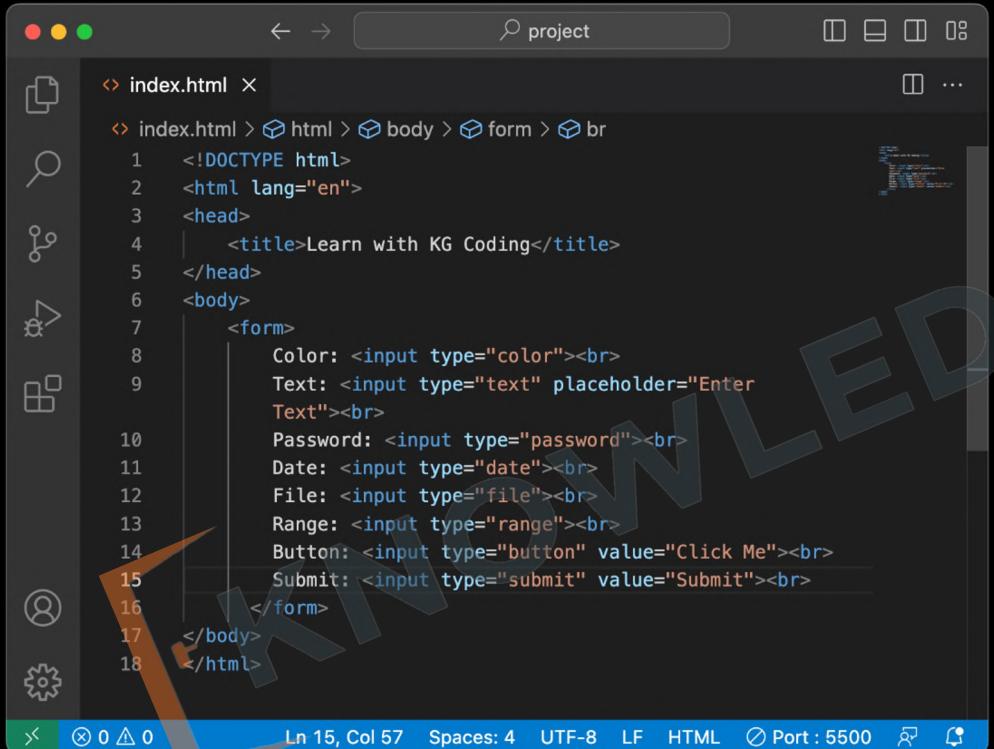


A screenshot of a web browser showing the rendered HTML from the code editor. The page displays the following inputs:

- Color: A color picker.
- Text: An input field with placeholder text "Enter Text".
- Password: A password input field.
- Date: A date input field with a placeholder "dd/mm/yyyy".
- File: A file input field showing "Choose file" and "No file chosen".
- Range: A range slider with a blue handle.
- Button: A button labeled "Click Me".

The browser window has a watermark "KNOWLEDGE GATE" across it.

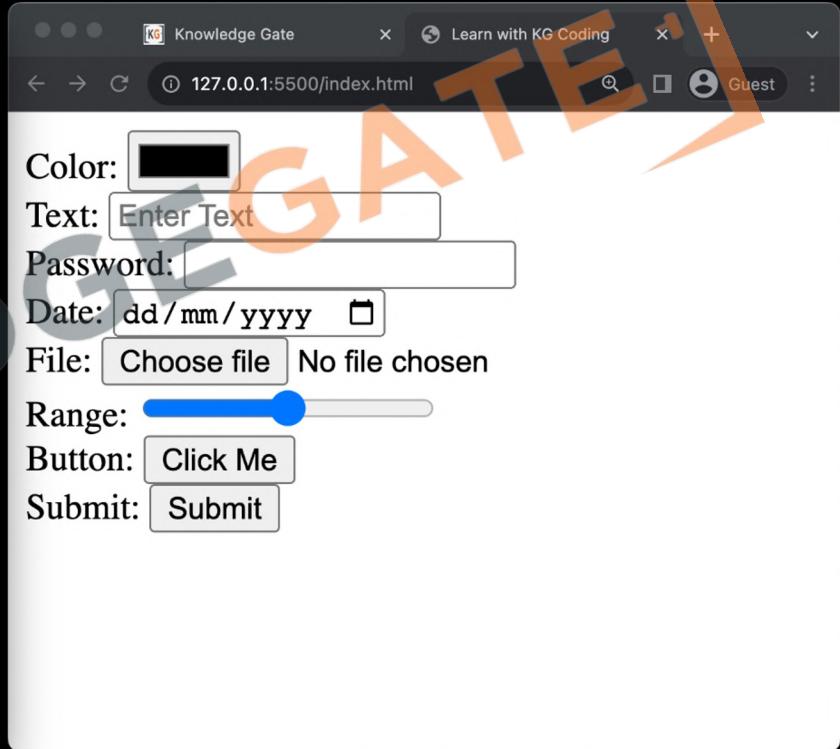
# 3.5 Input type: Submit



A screenshot of a code editor showing the file `index.html`. The code defines a form with various input fields:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        Color: <input type="color"><br>
        Text: <input type="text" placeholder="Enter Text"><br>
        Password: <input type="password"><br>
        Date: <input type="date"><br>
        File: <input type="file"><br>
        Range: <input type="range"><br>
        Button: <input type="button" value="Click Me"><br>
        Submit: <input type="submit" value="Submit"><br>
    </form>
</body>
</html>
```

The code editor has a dark theme with icons on the left. The status bar at the bottom shows: Ln 15, Col 57, Spaces: 4, UTF-8, LF, HTML, Port: 5500.

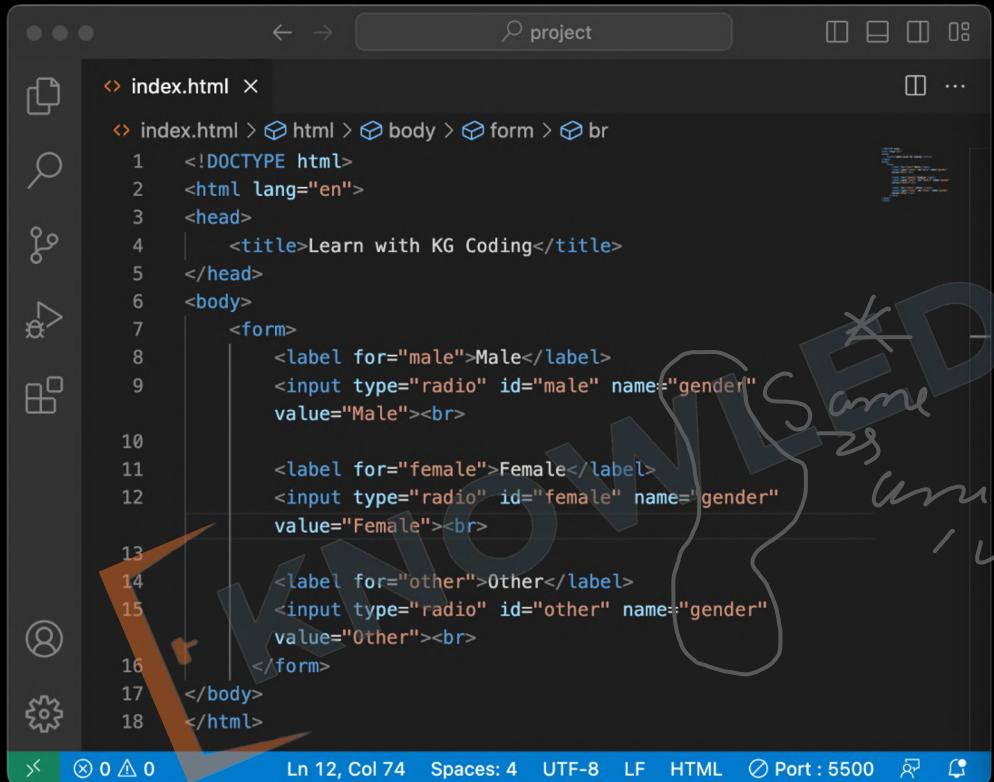


A screenshot of a web browser showing the rendered form from `index.html`. The browser window title is "Knowledge Gate". The form contains the following elements:

- Color: A color picker button.
- Text: An input field with placeholder text "Enter Text".
- Password: An input field for entering a password.
- Date: A date input field with a placeholder "dd/mm/yyyy" and a calendar icon.
- File: An input field for selecting a file, showing "Choose file" and "No file chosen".
- Range: A range slider with a blue handle.
- Button: A button labeled "Click Me".
- Submit: A button labeled "Submit".

The browser interface includes a search bar, tabs, and a guest user indicator.

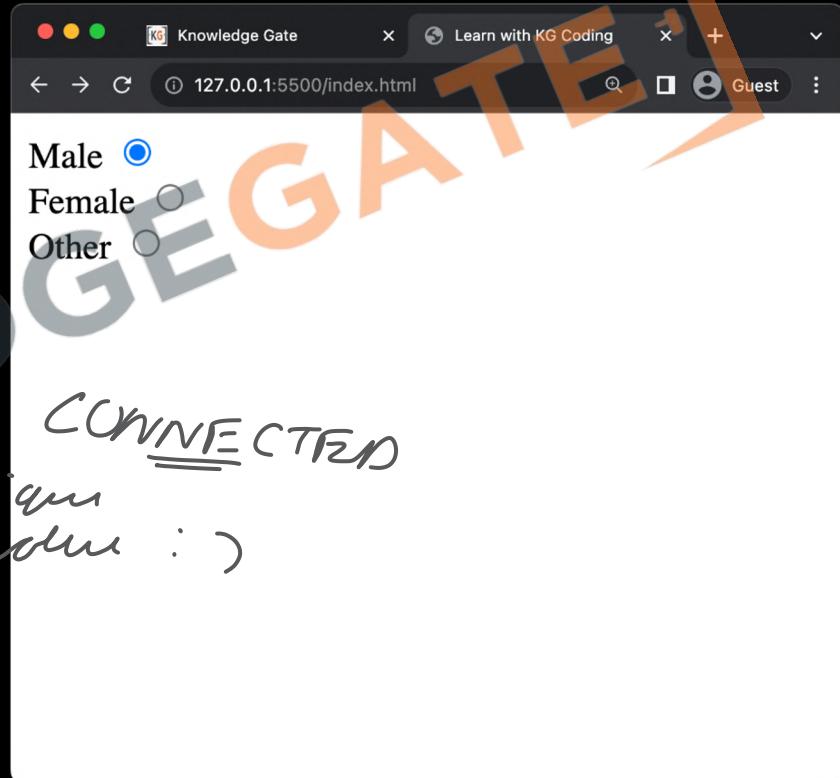
# 3.5 Input type: Radio



index.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Learn with KG Coding</title>
5 </head>
6 <body>
7   <form>
8     <label for="male">Male</label>
9     <input type="radio" id="male" name="gender" value="Male"><br>
10
11    <label for="female">Female</label>
12    <input type="radio" id="female" name="gender" value="Female"><br>
13
14    <label for="other">Other</label>
15    <input type="radio" id="other" name="gender" value="Other"><br>
16  </form>
17 </body>
18 </html>
```

Ln 12, Col 74 Spaces: 4 UTF-8 LF HTML ⚡ Port : 5500



Knowledge Gate

Learn with KG Coding

Guest

Male

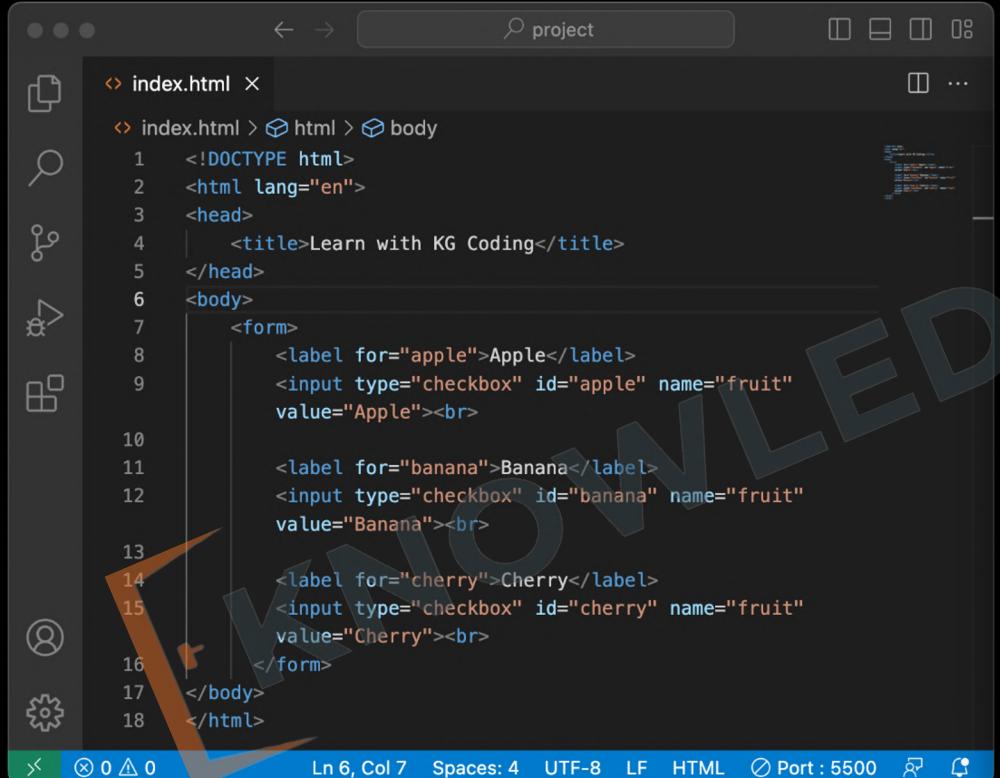
Female

Other

CONNECTED

Same  
as  
earlier  
/ value : )

# 3.5 Input type: Checkbox



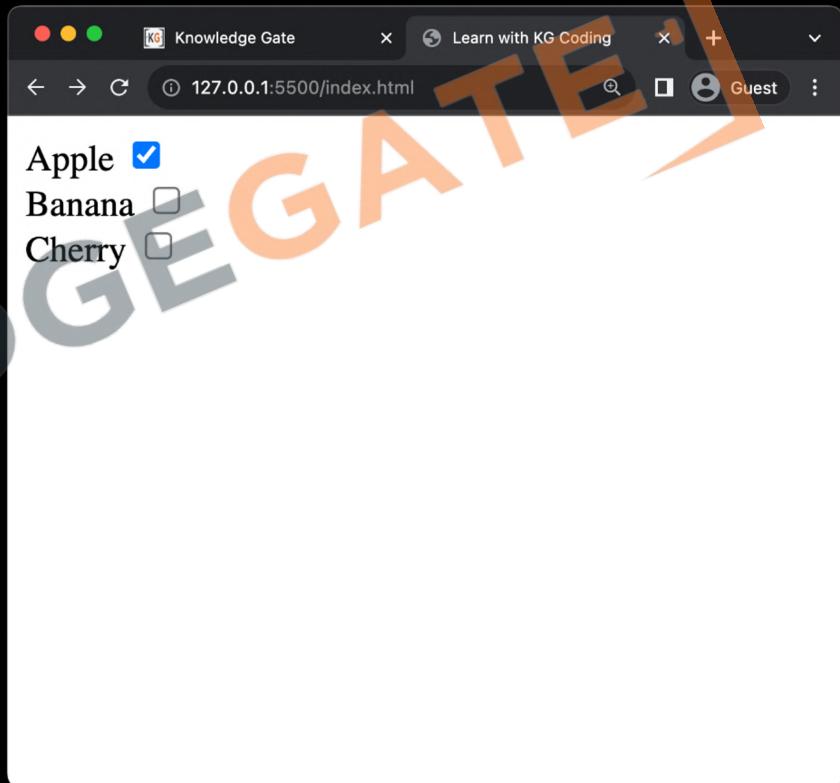
A screenshot of a code editor showing the file `index.html`. The code defines a form with three checkbox inputs, each associated with a fruit label:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        <label for="apple">Apple</label>
        <input type="checkbox" id="apple" name="fruit" value="Apple"><br>

        <label for="banana">Banana</label>
        <input type="checkbox" id="banana" name="fruit" value="Banana"><br>

        <label for="cherry">Cherry</label>
        <input type="checkbox" id="cherry" name="fruit" value="Cherry"><br>
    </form>
</body>
</html>
```

The code editor interface includes a sidebar with various icons and a status bar at the bottom.

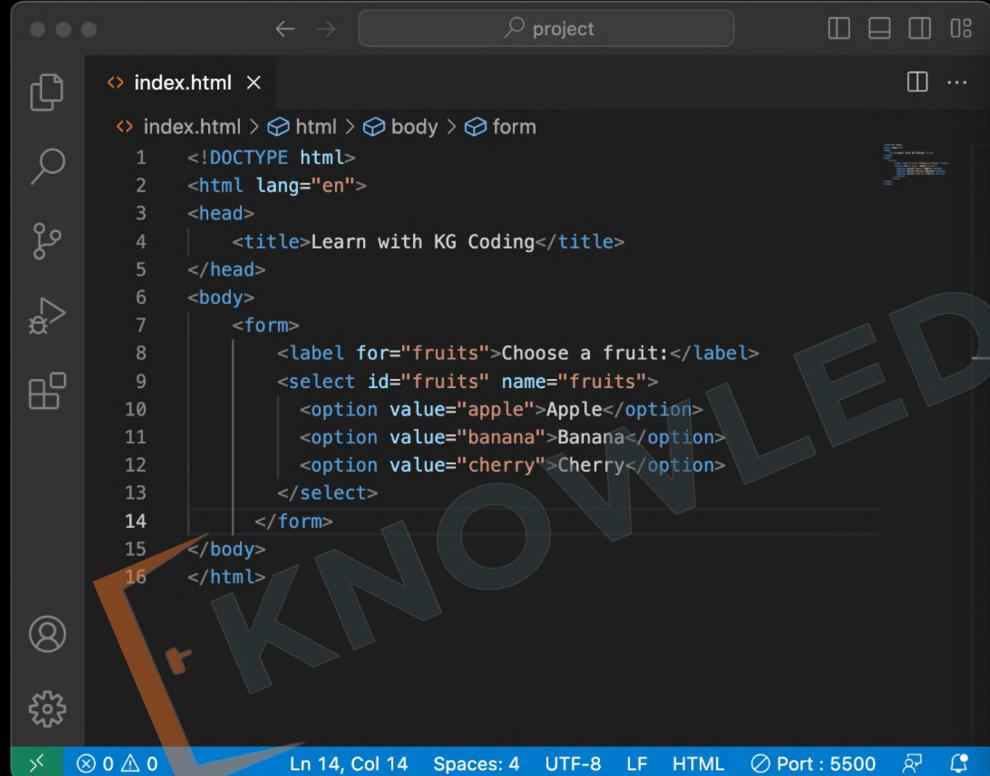


A screenshot of a web browser window titled "Knowledge Gate". The URL is `127.0.0.1:5500/index.html`. The page displays three items with checkboxes:

- Apple
- Banana
- Cherry

A large, semi-transparent watermark reading "KNOWLEDGE GATE" is overlaid across the browser window.

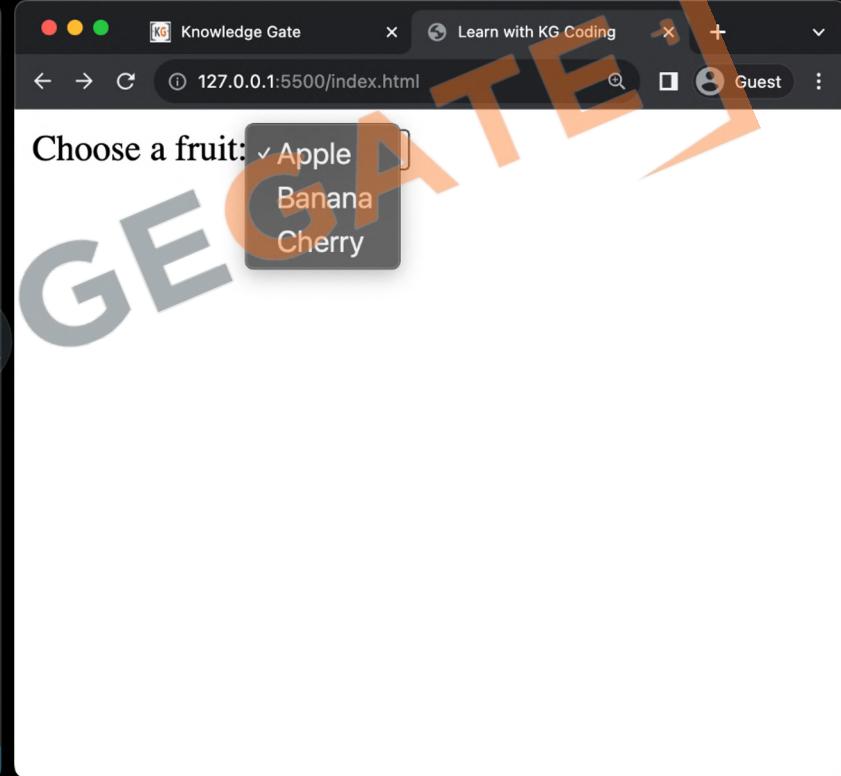
# 3.5 Input type: Select



A screenshot of a code editor showing the file `index.html`. The code defines a simple HTML form with a `select` element for selecting a fruit. The code is as follows:

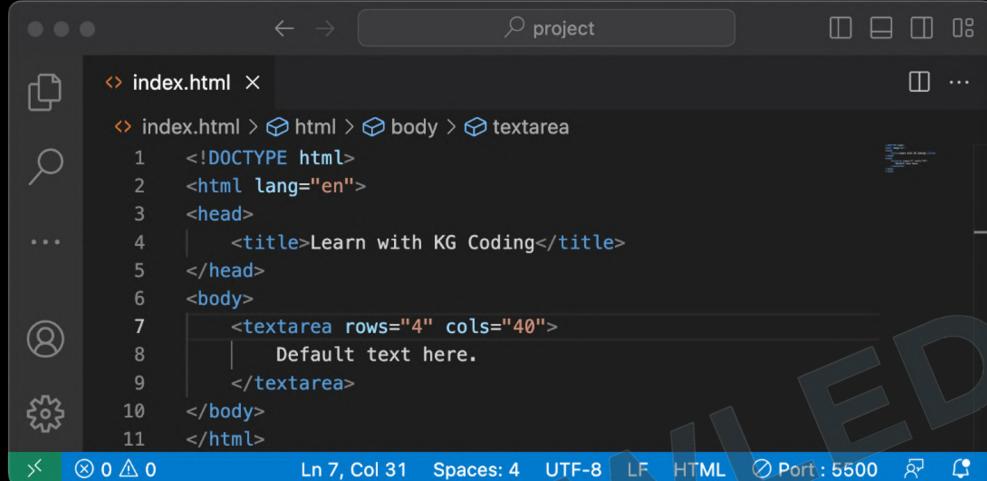
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <form>
        <label for="fruits">Choose a fruit:</label>
        <select id="fruits" name="fruits">
            <option value="apple">Apple</option>
            <option value="banana">Banana</option>
            <option value="cherry">Cherry</option>
        </select>
    </form>
</body>
</html>
```

The code editor interface includes a sidebar with various icons for file operations, a search bar, and a status bar at the bottom indicating line 14, column 14, and other file details.

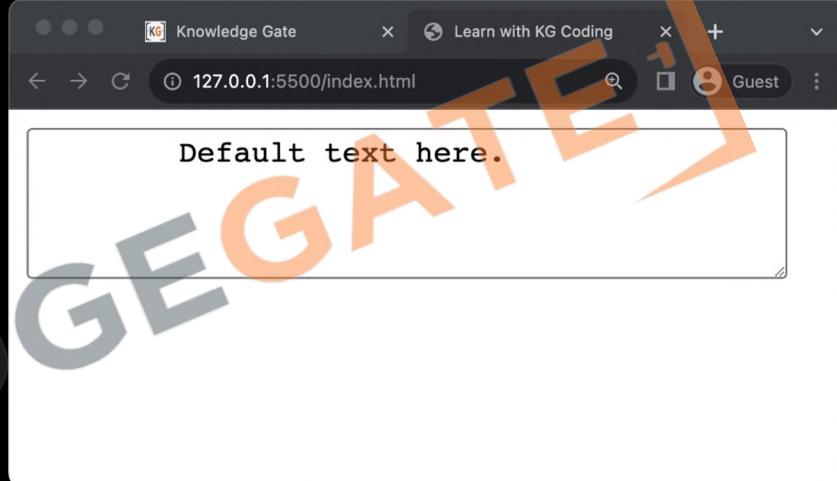


A screenshot of a web browser window titled "Knowledge Gate" showing the URL `127.0.0.1:5500/index.html`. The page content is "Choose a fruit:" followed by a dropdown menu with three options: "Apple", "Banana", and "Cherry". The word "KNOWLEDGE GATE" is overlaid diagonally across the browser window.

# 3.5 Input type: TextArea



```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <textarea rows="4" cols="40">
        Default text here.
    </textarea>
</body>
</html>
```



1. Purpose: `<textarea>` is used for multi-line text input in forms.
  1. **rows Property:** Specifies the visible number of lines in the textarea.
  2. **cols Property:** Sets the visible width measured in average character widths.
2. Resizable: Some browsers allow users to manually resize the textarea.

# Level 5

List, Tables & Forms



4GiFATE<sup>1</sup>  
Frame  
Tag

# 4.1 Using iFrames

The image shows a code editor on the left and a web browser on the right. The code editor displays the contents of index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Learn with KG Coding</title>
</head>
<body>
    <iframe width="300" height="200" src="https://en.wikipedia.org/wiki/Main_Page"></iframe>
</body>
</html>
```

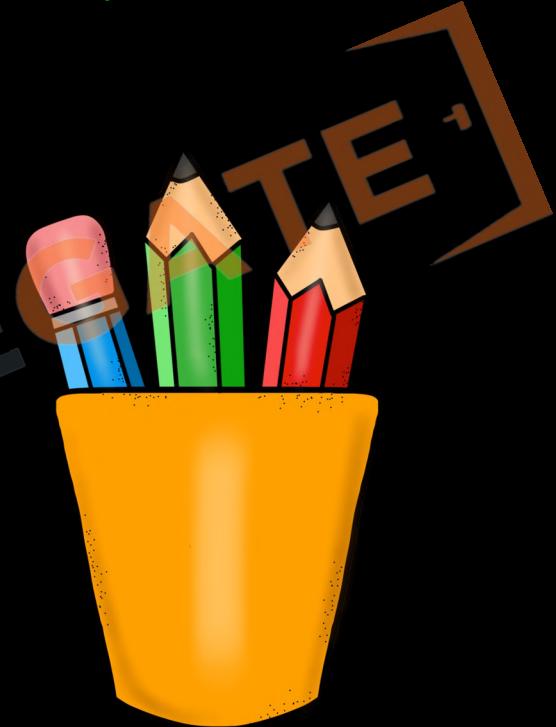
The browser window shows the rendered page with an orange watermark reading "ACKNOWLEDGED". It displays the Wikipedia homepage with the title "WIKIPEDIA The Free Encyclopedia" and links for "Main Page" and "Talk".

1. **Embedded Content:** Allows you to embed another webpage or multimedia content within a webpage.
2. **src Attribute:** Specifies the URL of the content to be embedded.
3. **Dimensions:** Width and height can be set using width and height attributes.

# Level 5 Revision

## List, Tables & Forms

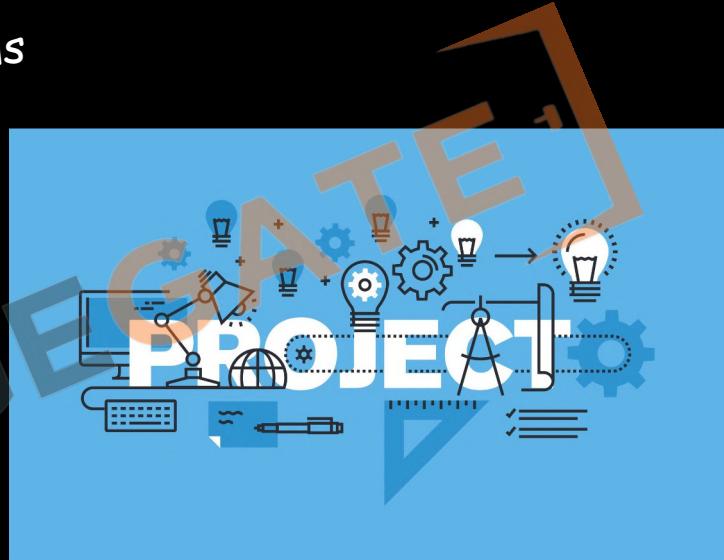
1. List Tag
  1. Ordered Lists
  2. Types of Ordered Lists
  3. Unordered Lists
2. Table Tag
  1. `<tr>`, `<td>`, `<th>` tags
  2. Captions
  3. Col spans
3. Forms
  1. Input tag
  2. Action Attributes
  3. Name and Value Property
  4. Label Tag
  5. Exploring Types
4. iFrame Tag
  1. Using iFrames



# Project Level 5

List, Tables & Forms

1. Create a **page** with all type of ordered list and one unordered list.
2. Create a **table** with headings, captions and a few rows. One of heading should take at least 3 columns.
3. Create a **contact me** form with relevant details for your resume website.
4. Use iFrame to add this video to your page.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)

# Level Bonus

Github Pages & CodeSpace

## 1. Github

1. What is Version Control
2. What is Git and GitHub
3. Account Creation
4. Creating a Repo
5. Creating a Codespace
6. Creating a Github page
7. Publishing our project

## 2. FrameWorks

1. React
2. Angular
3. Vue



# Level Bonus

Github Pages & CodeSpace

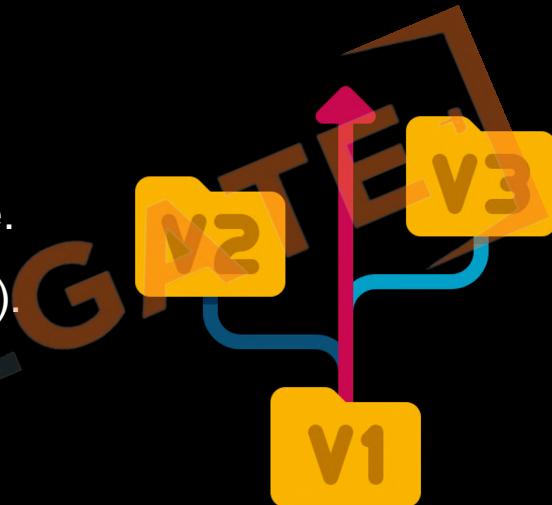


1.GitHub  
\*owned by microsoft

# 1.1 What is Version Control

\*Git is the Version Control system

1. **Definition:** A system to track changes in files over time.
2. **Types:** Centralized (like **SVN**) and Distributed (like **Git**).
3. **Purpose:** Helps in teamwork and fixes mistakes.
4. **Snapshots:** Each 'commit' saves a file version.
5. **Branching:** Lets you work on different tasks separately.
6. **Merge:** Combines changes from different people.
7. **Undo:** Easy to revert to older file versions.



# 1.2 What is Git and GitHub

Git is made by Linus,to Make Linux

Instagram of codes

## What is Git?

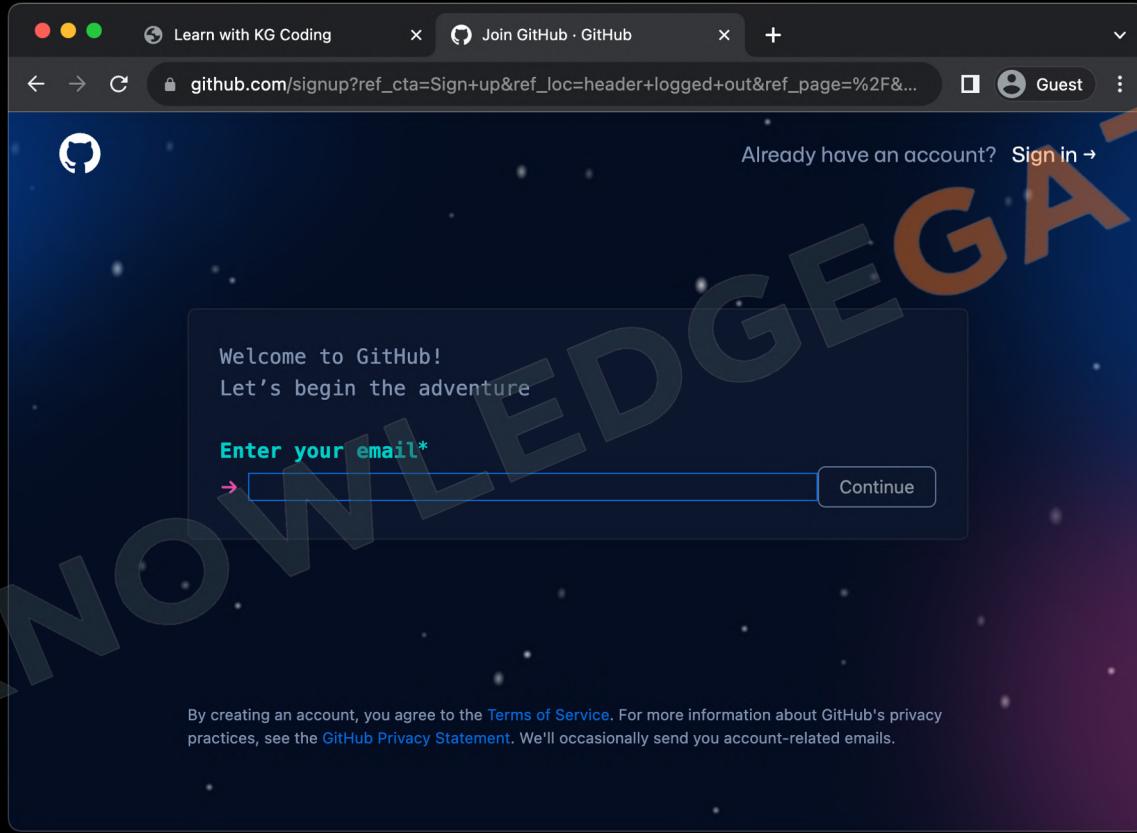
- **Definition:** A software tool that tracks changes in code, enabling collaboration and version control.
- **Commit:** Records a snapshot of file changes.
- **Branch:** Allows separate paths of development.
- **Merge:** Combines changes from different branches.

## What is GitHub?

- **Definition:** A web service for hosting and collaborating on Git repositories. BUT content to be PUBLIC
- **Fork:** Creates a personal copy of another user's repository.
- **Pull Request:** A way to propose changes to existing code.
- **Issues:** Used for tracking bugs and feature ideas.

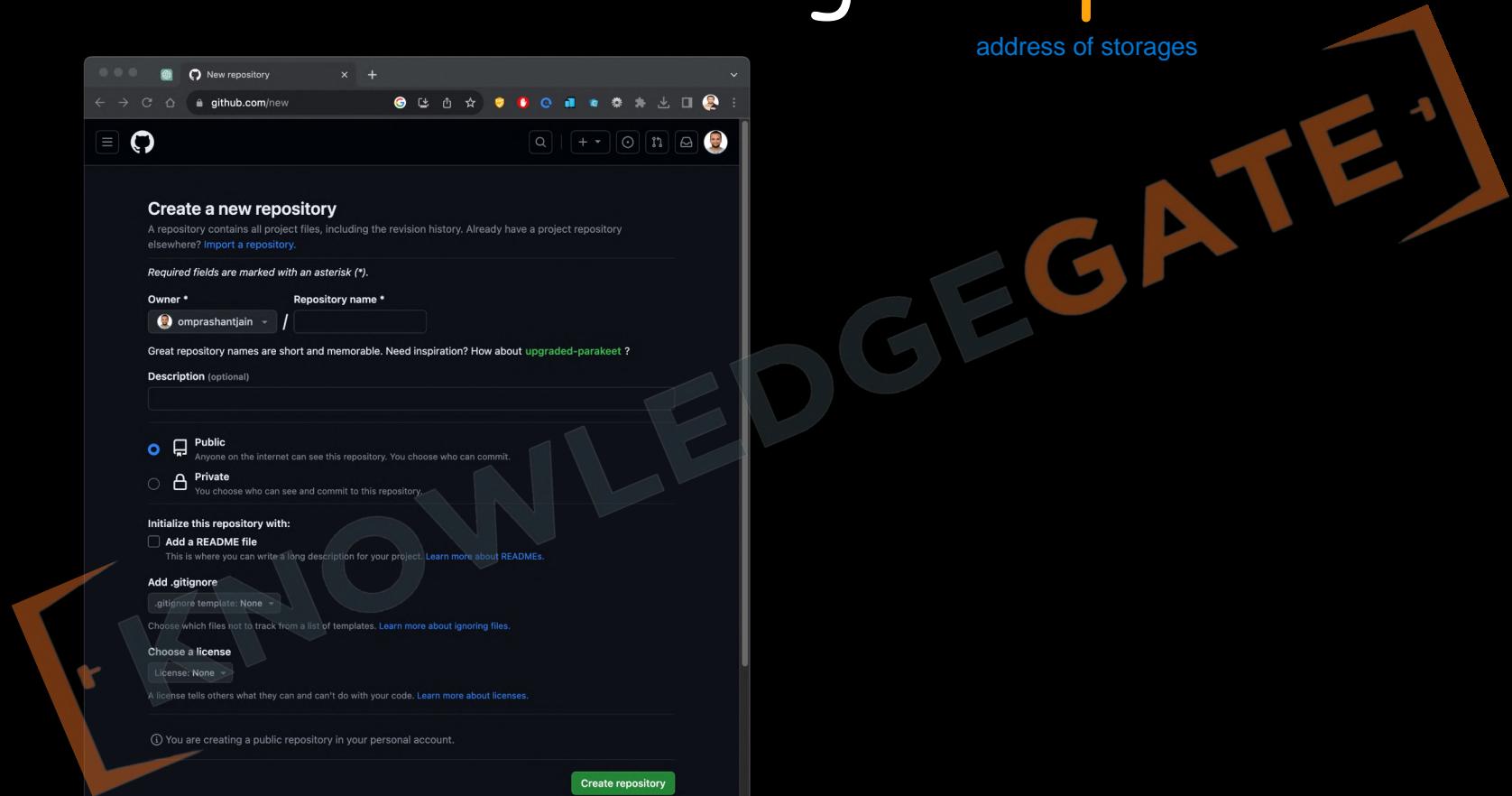
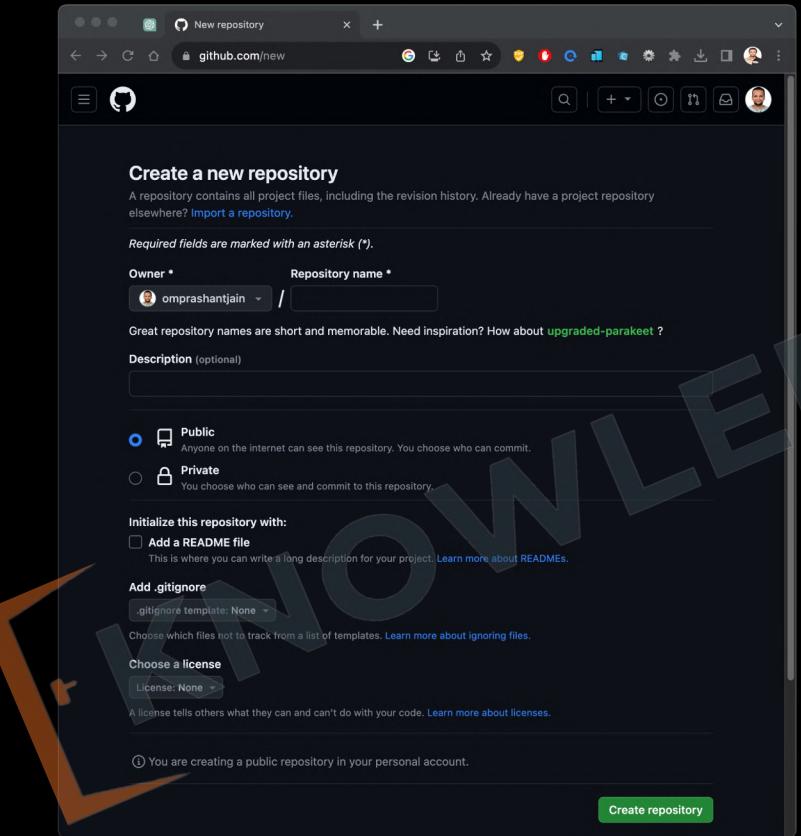


# 1.3 Account Creation



# 1.4 Creating a Repo

address of storages



# 1.5 Creating a CodeSpace

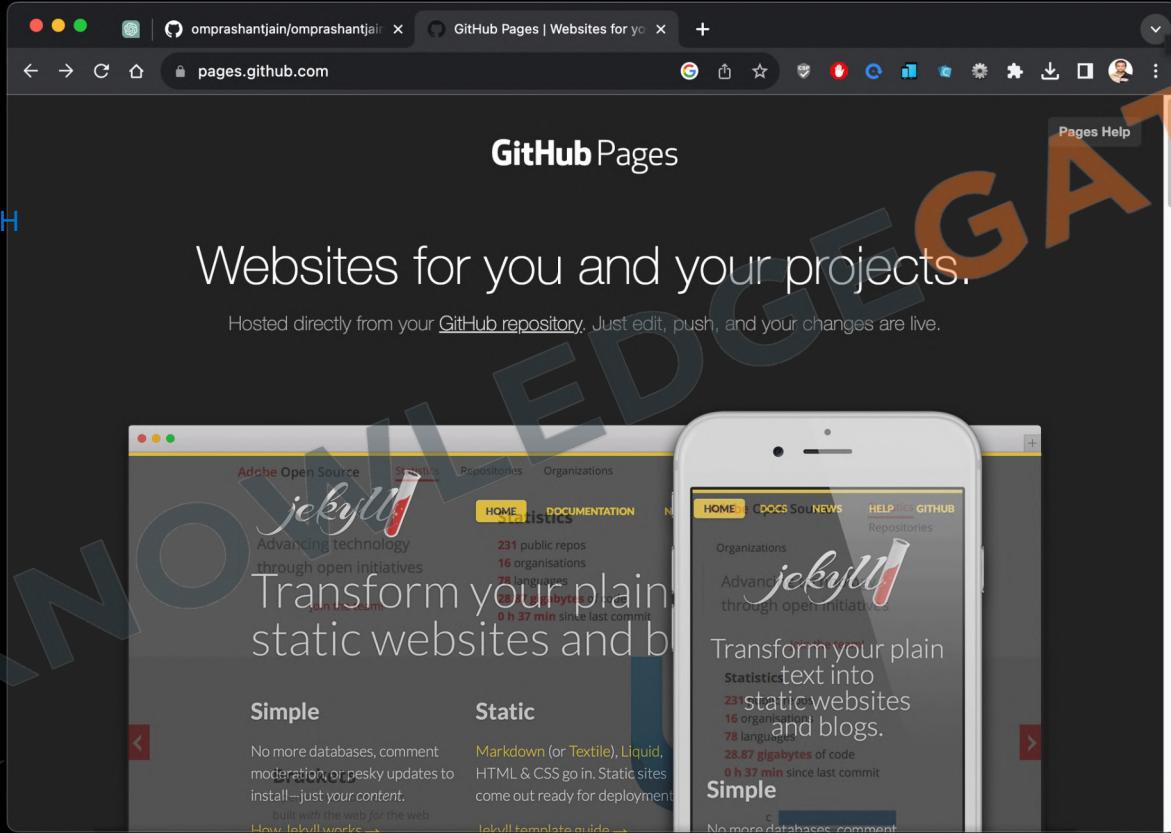
VSC ON WEB

A screenshot of a GitHub repository page for "KnowledgeGateCoding/java\_for\_beginner". The repository has 1 branch and 0 tags. The main file listed is "README.md". A modal window titled "Codespaces" is open, showing the message "No codespaces" and a button "Create codespace on main". The GitHub interface includes a navigation bar with "Code", "Issues", "Pull requests", "Discussions", "Actions", "Projects", "Wiki", "Security", and "Insights". The "Code" tab is selected. The "About" section on the right indicates "No description, website, or topics provided". Other details shown include "Readme", "Activity", "2 stars", "1 watching", "0 forks", and a "Report repository" link.

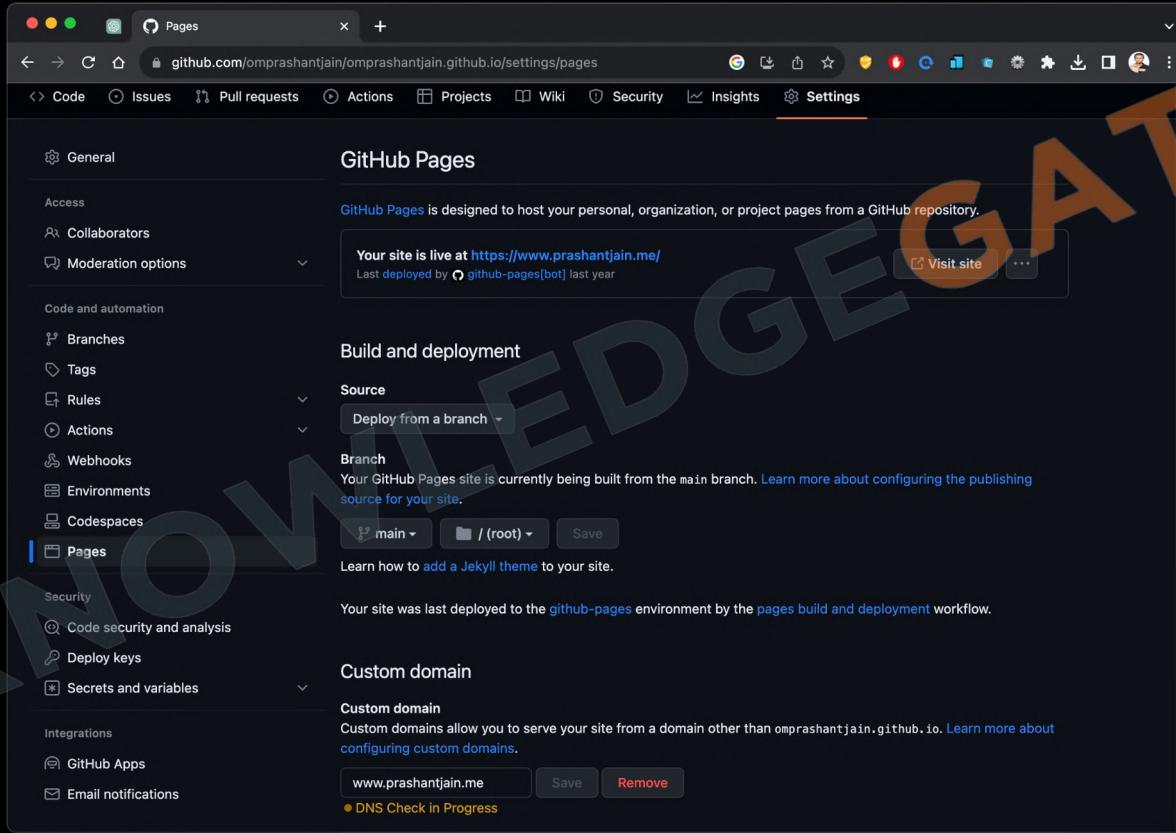
# 1.6 Creating a Github Page

\*for static pages only  
:(

MAIN=MASTER BRANCH



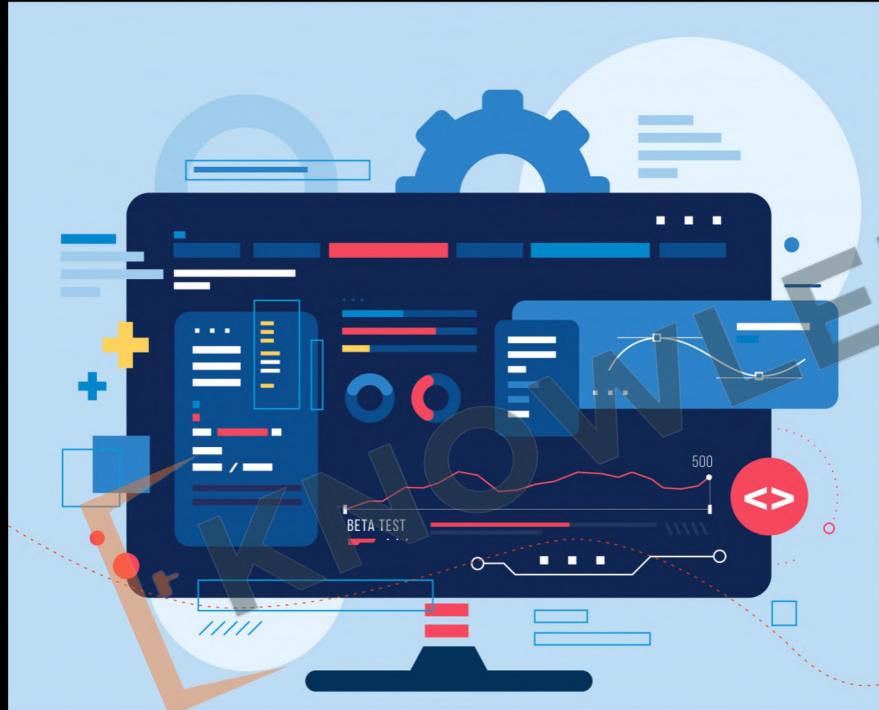
# 1.7 Publishing our Project



The screenshot shows the GitHub Pages settings page for the repository `omprashantjain/omprashantjain.github.io`. The main heading is "GitHub Pages". A message states: "GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository." Below this, it says "Your site is live at <https://www.prashantjain.me/>". A button labeled "Visit site" is present. The left sidebar lists various settings sections: General, Access, Collaborators, Moderation options, Code and automation (Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, Pages), Security (Code security and analysis, Deploy keys, Secrets and variables), Integrations (GitHub Apps, Email notifications). The "Pages" section is currently selected. In the main content area, under "Build and deployment", there is a "Source" dropdown set to "Deploy from a branch" with "main" selected. Under "Branch", it says "Your GitHub Pages site is currently being built from the `main` branch". There is a "Save" button. Below this, a link says "Learn how to add a Jekyll theme to your site." A note at the bottom states: "Your site was last deployed to the `github-pages` environment by the `pages` build and deployment workflow." Under "Custom domain", there is a field with "www.prashantjain.me", a "Save" button, and a "Remove" button. A note says: "Custom domains allow you to serve your site from a domain other than `omprashantjain.github.io`. Learn more about configuring custom domains." A status message at the bottom indicates: "● DNS Check in Progress".

# Level Bonus

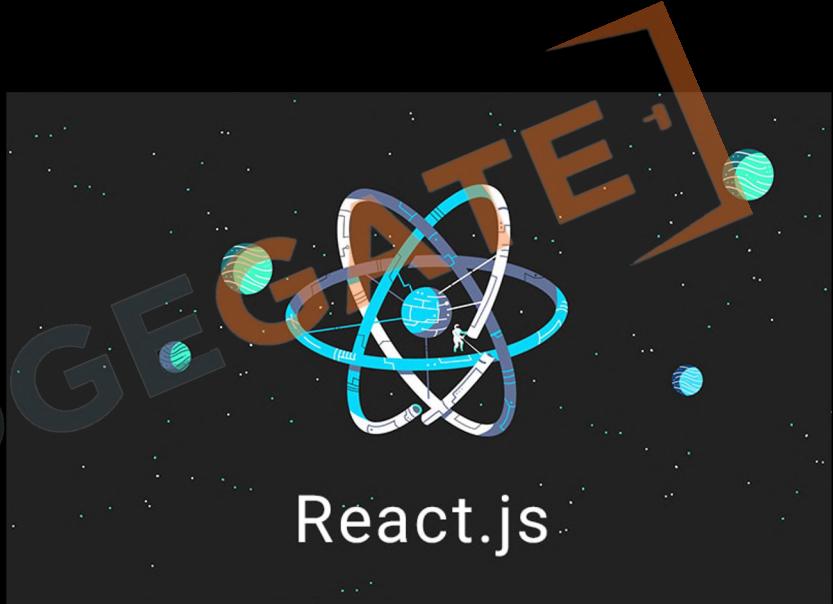
Github Pages & CodeSpace



EDGEGATE<sup>1</sup>  
2. Frameworks

# 3.1 ReactJS

1. **Definition:** A tool for making websites interactive. [LESS CODE](#)
2. **Components:** Reusable pieces for building a webpage.
3. **Virtual DOM:** Makes websites faster by updating only what's needed.
4. **JSX:** A special way to write code that looks like [HTML](#).
5. **State:** Keeps track of changes on the webpage.
6. **Props:** Shares information between different parts of a webpage.



## 3.2 AngularJS

1. **Definition:** A framework for building web applications, developed by **Google**.
2. **Two-Way Data Binding:** Updates both the view and the model simultaneously.
3. **Directives:** Custom **HTML** tags for added functionality.
4. **Dependency Injection:** **Automatically manages** how parts of the app work together.
5. **Controllers:** Manages the data for a specific part of the webpage.
6. **SPA Support:** Good for **Single Page Applications** where the page doesn't reload.



### 3.3 VueJS

1. **Definition:** A JavaScript framework for creating web interfaces.
2. **Components:** Small, reusable parts for building a website.
3. **Reactivity:** Automatically updates the webpage when data changes.
4. **Directives:** Special tokens in HTML for added functionality.
5. **Vuex:** Helps manage shared data across the site.
6. **Single-File Components:** Keeps template, script, and style in one file.



# Level Bonus Revision

Github Pages & CodeSpace

## 1. Github

1. What is Version Control
2. What is Git and GitHub
3. Account Creation
4. Creating a Repo
5. Creating a Codespace
6. Creating a Github page
7. Publishing our project

## 2. FrameWorks

1. React
2. Angular
3. Vue

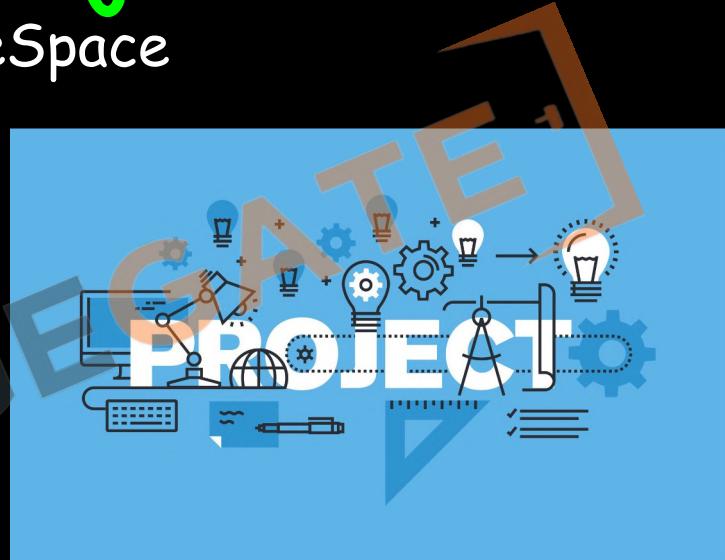


ACKNOWLEDGE

# Level Bonus Project

Github Pages & CodeSpace

1. Create a Github account if you don't have one.
2. Create a repo for the project you have done till now.
3. Create a Codespace and make changes.
4. Publish it using Github pages.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)

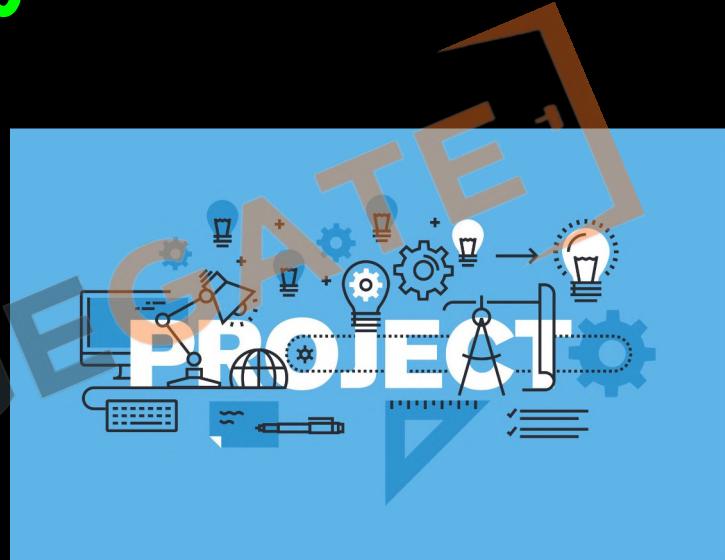


[Sanchit Socket](#)

# Major Project

Idea

1. Create your **Resume** or Portfolio website.
2. Create a repo in **Github**.
3. Publish it on **Github pages**.
4. Add the link to your **resume**.
5. Add the link in the **comment** section.



# KG Coding

Some Other One shot Video Links:

- [Complete HTML](#)
- [Complete CSS](#)
- [Complete JavaScript](#)
- [Complete React and Redux](#)
- [One shot University Exam Series](#)

<http://www.kgcoding.in/>

Our  YouTube Channels

[KG Coding Android App](#)



[KG Coding](#)



[Knowledge GATE](#)



[KG Placement Prep](#)



[Sanchit Socket](#)