

DoctorAi

doctoraibyanshul.netlify.app

Problem Statement

- In today's digital age, individuals frequently search online for health-related information, leading to misinformation, self-diagnosis, and delayed professional care. Access to accurate and personalized health guidance remains limited, especially in rural areas.

Objective / Aim

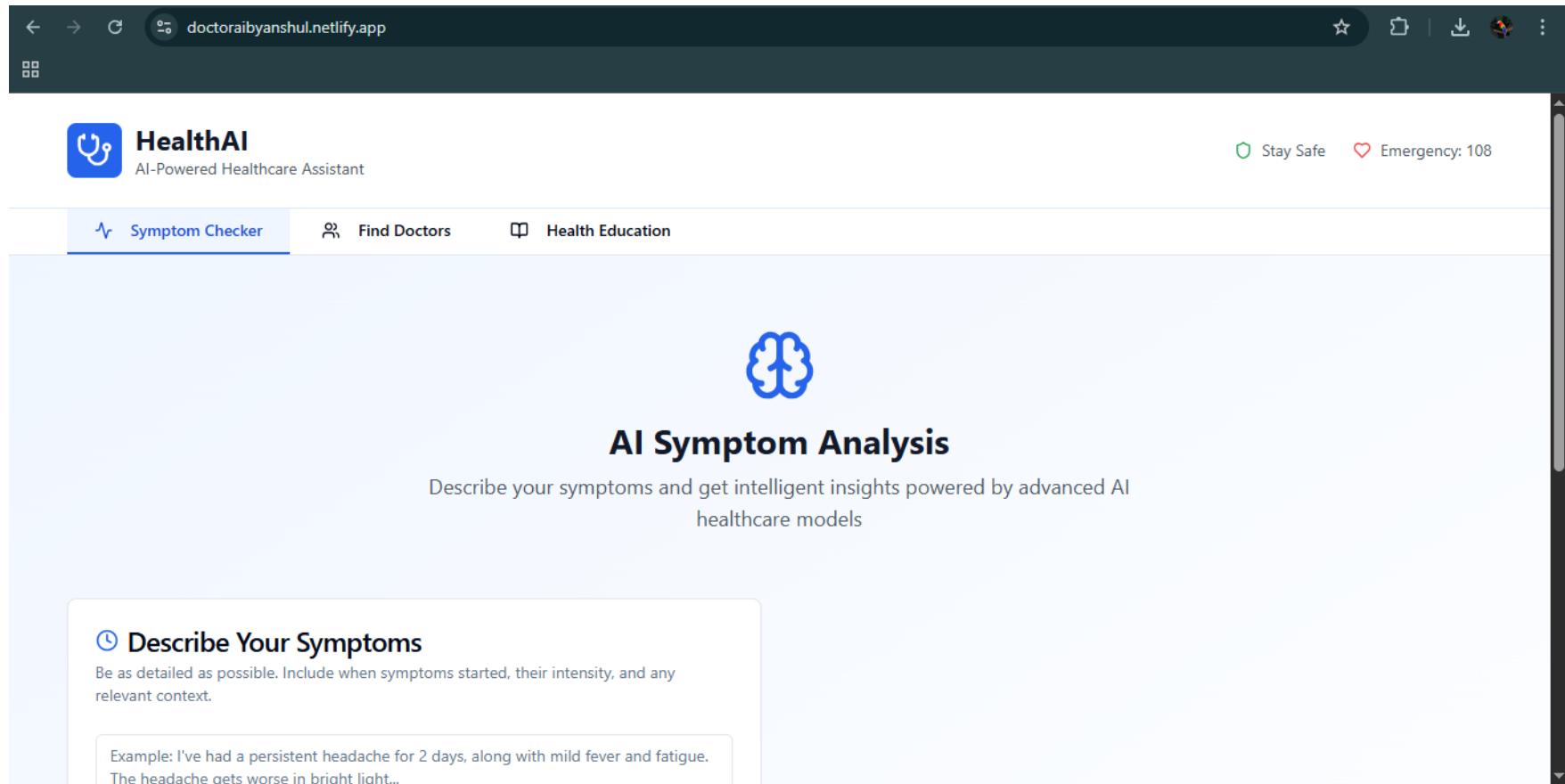
- DoctorAI aims to build an intelligent AI-powered system that understands users' health queries and provides verified insights, guiding them toward next steps such as consulting doctors.

Proposed Solution / Approach

- Uses NLP and LLMs to interpret user queries.
- Integrates curated medical databases.
- Provides reliable, explainable responses.
- Supports classification of symptoms and real-time triage.
- Future additions: multilingual support, wearable data.

Expected Impact

- Acts as a virtual pre-diagnosis assistant.
- Reduces misinformation.
- Supports overburdened healthcare systems.
- Enhances accessibility in rural and semi-urban regions.



The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The top navigation bar includes File, Edit, Selection, View, Go, Run, and a search bar. The left sidebar contains various icons for file operations like Open, Save, Find, and Debug.

The main editor area has tabs for JS server.js, JS yellow.js (which is currently active), JS Web3Wallet.js 1, and JS App.js 1. The code in yellow.js is as follows:

```
C:\> Users > hp > OneDrive > Desktop > ansh02 > bhilare_projects > doctor-ai-anshul-main > utils > JS yellow.js > ...
36     export function initYellowSession(appId, signer) {
37         ws.on("message", (msg) => {
38             ...
39         });
40         ws.on("close", () => console.log("🟡 Connection closed"));
41         ws.on("error", (err) => console.error("🟡 Error:", err));
42         ...
43     }
44     return ws;
45 }
46 }
```

The bottom section features a terminal tab labeled TERMINAL. The terminal output shows the following command and its results:

```
PS C:\Users\hp\OneDrive\Desktop\ansh02\bhilare_projects\doctor-ai-anshul-main> npm install web3
>>
To address all issues, run:
  npm audit fix

Run `npm audit` for details.
② PS C:\Users\hp\OneDrive\Desktop\ansh02\bhilare_projects\doctor-ai-anshul-main> node server.js
[dotenv@17.2.3] injecting env (3) from .env -- tip: ⚠ write to custom object with { processEnv: myObject }
(node:17300) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:17300) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version
  Doctor AI + Yellow SDK backend running on port 5000
  MongoDB connected for Doctor AI
  Connected to Yellow Network
C:\Users\hp\OneDrive\Desktop\ansh02\bhilare_projects\doctor-ai-anshul-main\node_modules\@erc724\nitrolite\dist\rpc\nitrolite.js:28
```

The status bar at the bottom displays the current file path, line number (Ln 57), column number (Col 1), and encoding (UTF-8). It also shows tabs for Launchpad, Connect, Live Share, and Quokka, along with other standard status indicators.

File Edit Selection View Go Run ... ⏪ ⏩ Search ...

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. [Manage](#) [Learn More](#)

EXPLORER OPEN EDITORS TIMELINE MAVEN

Welcome JS yellow.js JS server.js JS yellowTest.js

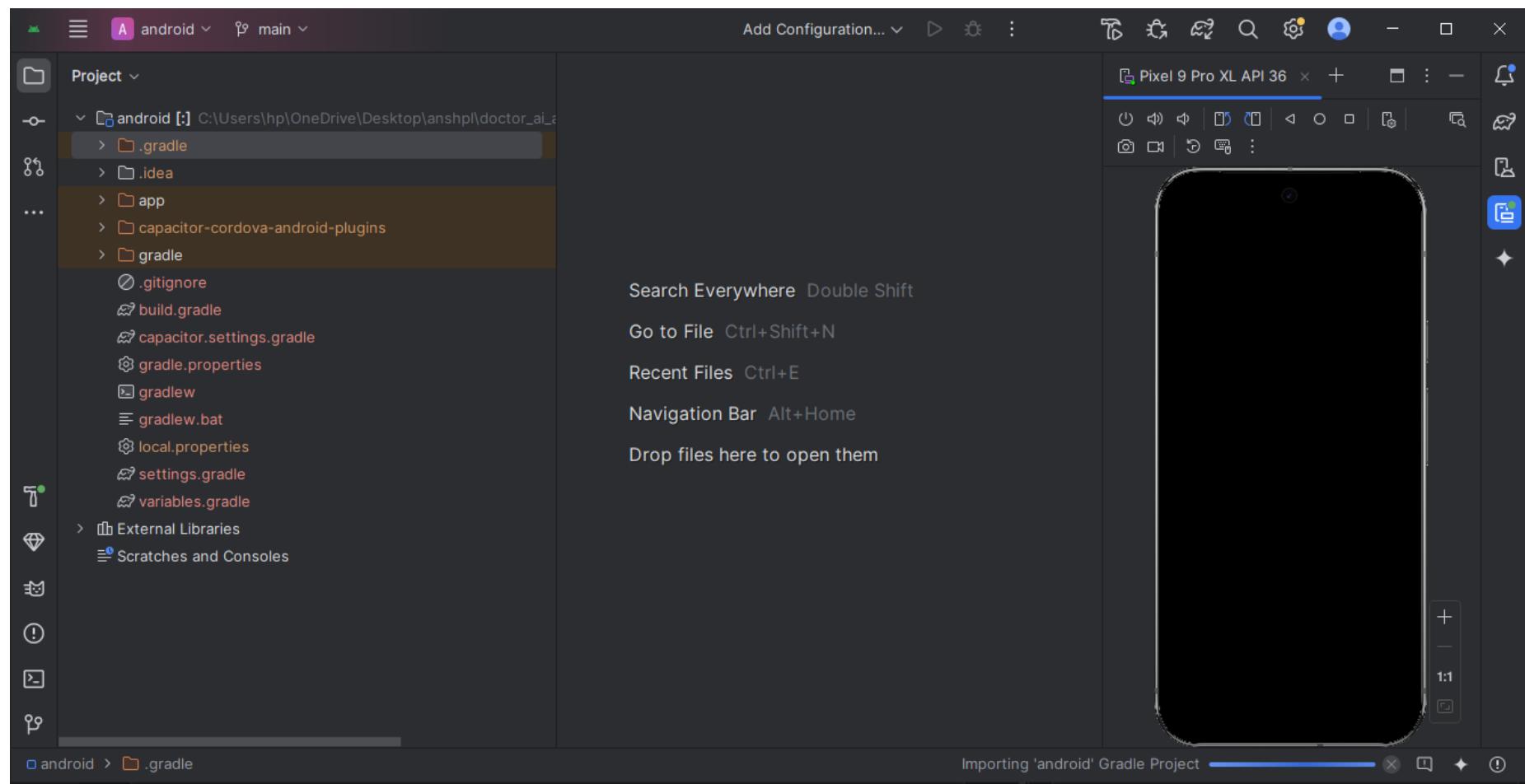
C:\> Users > hp > OneDrive > Desktop > ansh02 > bhilare_projects > doctor-ai-anshul-main > JS yellowTest.js > ...

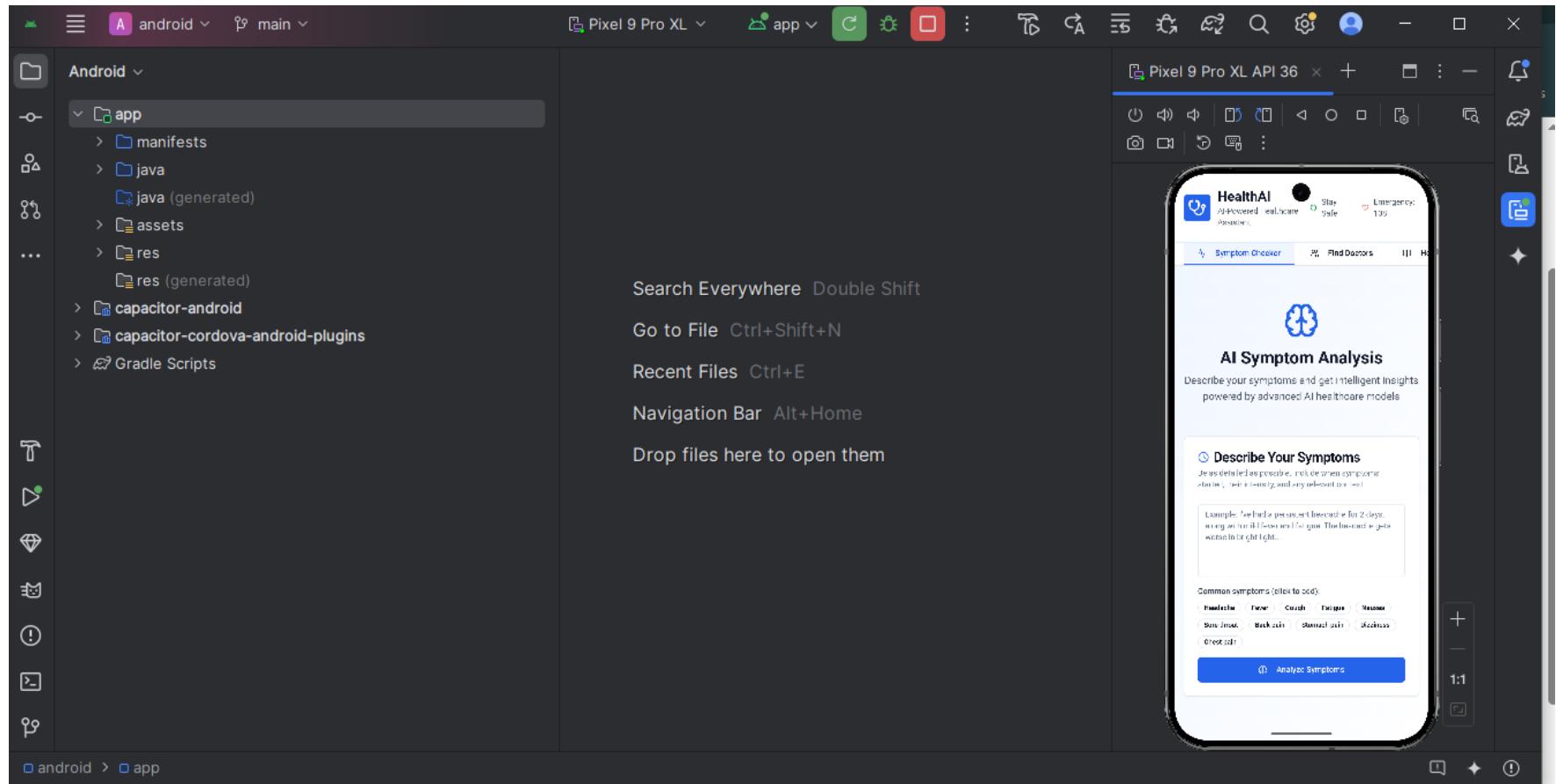
```
1 import { WEB_SOCKET } from './constants';
2
3 const ws = new WebSocket("wss://clearnet.yellow.com/ws");
4
5 ws.on("open", () => {
6     console.log("Connected to Yellow Network (Test)");
7
8     // Send a test RPC request that does NOT require signing
9     const testMsg = {
10         method: "eth_blockNumber", // fetch latest block
11         params: [],
12         id: 1,
13     };
14
15     ws.send(JSON.stringify(testMsg));
16
17     ws.on("message", (data) => {
18         const response = JSON.parse(data);
19
20         if (response.error) {
21             console.error(`RPC error: ${response.error}`);
22             return;
23         }
24
25         console.log(`RPC response: ${JSON.stringify(response)}`);
26     });
27
28     ws.on("close", () => {
29         console.log("Connection closed");
30     });
31
32     ws.on("error", (err) => {
33         console.error(`WebSocket error: ${err}`);
34     });
35
36     ws.on("message", (data) => {
37         const response = JSON.parse(data);
38
39         if (response.error) {
40             console.error(`RPC error: ${response.error}`);
41             return;
42         }
43
44         console.log(`RPC response: ${JSON.stringify(response)}`);
45     });
46
47     ws.on("close", () => {
48         console.log("Connection closed");
49     });
50
51     ws.on("error", (err) => {
52         console.error(`WebSocket error: ${err}`);
53     });
54
55     ws.on("message", (data) => {
56         const response = JSON.parse(data);
57
58         if (response.error) {
59             console.error(`RPC error: ${response.error}`);
60             return;
61         }
62
63         console.log(`RPC response: ${JSON.stringify(response)}`);
64     });
65
66     ws.on("close", () => {
67         console.log("Connection closed");
68     });
69
70     ws.on("error", (err) => {
71         console.error(`WebSocket error: ${err}`);
72     });
73
74     ws.on("message", (data) => {
75         const response = JSON.parse(data);
76
77         if (response.error) {
78             console.error(`RPC error: ${response.error}`);
79             return;
80         }
81
82         console.log(`RPC response: ${JSON.stringify(response)}`);
83     });
84
85     ws.on("close", () => {
86         console.log("Connection closed");
87     });
88
89     ws.on("error", (err) => {
90         console.error(`WebSocket error: ${err}`);
91     });
92
93     ws.on("message", (data) => {
94         const response = JSON.parse(data);
95
96         if (response.error) {
97             console.error(`RPC error: ${response.error}`);
98             return;
99         }
100
101         console.log(`RPC response: ${JSON.stringify(response)}`);
102     });
103
104     ws.on("close", () => {
105         console.log("Connection closed");
106     });
107
108     ws.on("error", (err) => {
109         console.error(`WebSocket error: ${err}`);
110     });
111
112     ws.on("message", (data) => {
113         const response = JSON.parse(data);
114
115         if (response.error) {
116             console.error(`RPC error: ${response.error}`);
117             return;
118         }
119
120         console.log(`RPC response: ${JSON.stringify(response)}`);
121     });
122
123     ws.on("close", () => {
124         console.log("Connection closed");
125     });
126
127     ws.on("error", (err) => {
128         console.error(`WebSocket error: ${err}`);
129     });
130
131     ws.on("message", (data) => {
132         const response = JSON.parse(data);
133
134         if (response.error) {
135             console.error(`RPC error: ${response.error}`);
136             return;
137         }
138
139         console.log(`RPC response: ${JSON.stringify(response)}`);
140     });
141
142     ws.on("close", () => {
143         console.log("Connection closed");
144     });
145
146     ws.on("error", (err) => {
147         console.error(`WebSocket error: ${err}`);
148     });
149
150     ws.on("message", (data) => {
151         const response = JSON.parse(data);
152
153         if (response.error) {
154             console.error(`RPC error: ${response.error}`);
155             return;
156         }
157
158         console.log(`RPC response: ${JSON.stringify(response)}`);
159     });
160
161     ws.on("close", () => {
162         console.log("Connection closed");
163     });
164
165     ws.on("error", (err) => {
166         console.error(`WebSocket error: ${err}`);
167     });
168
169     ws.on("message", (data) => {
170         const response = JSON.parse(data);
171
172         if (response.error) {
173             console.error(`RPC error: ${response.error}`);
174             return;
175         }
176
177         console.log(`RPC response: ${JSON.stringify(response)}`);
178     });
179
180     ws.on("close", () => {
181         console.log("Connection closed");
182     });
183
184     ws.on("error", (err) => {
185         console.error(`WebSocket error: ${err}`);
186     });
187
188     ws.on("message", (data) => {
189         const response = JSON.parse(data);
190
191         if (response.error) {
192             console.error(`RPC error: ${response.error}`);
193             return;
194         }
195
196         console.log(`RPC response: ${JSON.stringify(response)}`);
197     });
198
199     ws.on("close", () => {
200         console.log("Connection closed");
201     });
202
203     ws.on("error", (err) => {
204         console.error(`WebSocket error: ${err}`);
205     });
206
207     ws.on("message", (data) => {
208         const response = JSON.parse(data);
209
210         if (response.error) {
211             console.error(`RPC error: ${response.error}`);
212             return;
213         }
214
215         console.log(`RPC response: ${JSON.stringify(response)}`);
216     });
217
218     ws.on("close", () => {
219         console.log("Connection closed");
220     });
221
222     ws.on("error", (err) => {
223         console.error(`WebSocket error: ${err}`);
224     });
225
226     ws.on("message", (data) => {
227         const response = JSON.parse(data);
228
229         if (response.error) {
230             console.error(`RPC error: ${response.error}`);
231             return;
232         }
233
234         console.log(`RPC response: ${JSON.stringify(response)}`);
235     });
236
237     ws.on("close", () => {
238         console.log("Connection closed");
239     });
240
241     ws.on("error", (err) => {
242         console.error(`WebSocket error: ${err}`);
243     });
244
245     ws.on("message", (data) => {
246         const response = JSON.parse(data);
247
248         if (response.error) {
249             console.error(`RPC error: ${response.error}`);
250             return;
251         }
252
253         console.log(`RPC response: ${JSON.stringify(response)}`);
254     });
255
256     ws.on("close", () => {
257         console.log("Connection closed");
258     });
259
260     ws.on("error", (err) => {
261         console.error(`WebSocket error: ${err}`);
262     });
263
264     ws.on("message", (data) => {
265         const response = JSON.parse(data);
266
267         if (response.error) {
268             console.error(`RPC error: ${response.error}`);
269             return;
270         }
271
272         console.log(`RPC response: ${JSON.stringify(response)}`);
273     });
274
275     ws.on("close", () => {
276         console.log("Connection closed");
277     });
278
279     ws.on("error", (err) => {
280         console.error(`WebSocket error: ${err}`);
281     });
282
283     ws.on("message", (data) => {
284         const response = JSON.parse(data);
285
286         if (response.error) {
287             console.error(`RPC error: ${response.error}`);
288             return;
289         }
290
291         console.log(`RPC response: ${JSON.stringify(response)}`);
292     });
293
294     ws.on("close", () => {
295         console.log("Connection closed");
296     });
297
298     ws.on("error", (err) => {
299         console.error(`WebSocket error: ${err}`);
300     });
301
302     ws.on("message", (data) => {
303         const response = JSON.parse(data);
304
305         if (response.error) {
306             console.error(`RPC error: ${response.error}`);
307             return;
308         }
309
310         console.log(`RPC response: ${JSON.stringify(response)}`);
311     });
312
313     ws.on("close", () => {
314         console.log("Connection closed");
315     });
316
317     ws.on("error", (err) => {
318         console.error(`WebSocket error: ${err}`);
319     });
320
321     ws.on("message", (data) => {
322         const response = JSON.parse(data);
323
324         if (response.error) {
325             console.error(`RPC error: ${response.error}`);
326             return;
327         }
328
329         console.log(`RPC response: ${JSON.stringify(response)}`);
330     });
331
332     ws.on("close", () => {
333         console.log("Connection closed");
334     });
335
336     ws.on("error", (err) => {
337         console.error(`WebSocket error: ${err}`);
338     });
339
340     ws.on("message", (data) => {
341         const response = JSON.parse(data);
342
343         if (response.error) {
344             console.error(`RPC error: ${response.error}`);
345             return;
346         }
347
348         console.log(`RPC response: ${JSON.stringify(response)}`);
349     });
350
351     ws.on("close", () => {
352         console.log("Connection closed");
353     });
354
355     ws.on("error", (err) => {
356         console.error(`WebSocket error: ${err}`);
357     });
358
359     ws.on("message", (data) => {
360         const response = JSON.parse(data);
361
362         if (response.error) {
363             console.error(`RPC error: ${response.error}`);
364             return;
365         }
366
367         console.log(`RPC response: ${JSON.stringify(response)}`);
368     });
369
370     ws.on("close", () => {
371         console.log("Connection closed");
372     });
373
374     ws.on("error", (err) => {
375         console.error(`WebSocket error: ${err}`);
376     });
377
378     ws.on("message", (data) => {
379         const response = JSON.parse(data);
380
381         if (response.error) {
382             console.error(`RPC error: ${response.error}`);
383             return;
384         }
385
386         console.log(`RPC response: ${JSON.stringify(response)}`);
387     });
388
389     ws.on("close", () => {
390         console.log("Connection closed");
391     });
392
393     ws.on("error", (err) => {
394         console.error(`WebSocket error: ${err}`);
395     });
396
397     ws.on("message", (data) => {
398         const response = JSON.parse(data);
399
400         if (response.error) {
401             console.error(`RPC error: ${response.error}`);
402             return;
403         }
404
405         console.log(`RPC response: ${JSON.stringify(response)}`);
406     });
407
408     ws.on("close", () => {
409         console.log("Connection closed");
410     });
411
412     ws.on("error", (err) => {
413         console.error(`WebSocket error: ${err}`);
414     });
415
416     ws.on("message", (data) => {
417         const response = JSON.parse(data);
418
419         if (response.error) {
420             console.error(`RPC error: ${response.error}`);
421             return;
422         }
423
424         console.log(`RPC response: ${JSON.stringify(response)}`);
425     });
426
427     ws.on("close", () => {
428         console.log("Connection closed");
429     });
430
431     ws.on("error", (err) => {
432         console.error(`WebSocket error: ${err}`);
433     });
434
435     ws.on("message", (data) => {
436         const response = JSON.parse(data);
437
438         if (response.error) {
439             console.error(`RPC error: ${response.error}`);
440             return;
441         }
442
443         console.log(`RPC response: ${JSON.stringify(response)}`);
444     });
445
446     ws.on("close", () => {
447         console.log("Connection closed");
448     });
449
450     ws.on("error", (err) => {
451         console.error(`WebSocket error: ${err}`);
452     });
453
454     ws.on("message", (data) => {
455         const response = JSON.parse(data);
456
457         if (response.error) {
458             console.error(`RPC error: ${response.error}`);
459             return;
460         }
461
462         console.log(`RPC response: ${JSON.stringify(response)}`);
463     });
464
465     ws.on("close", () => {
466         console.log("Connection closed");
467     });
468
469     ws.on("error", (err) => {
470         console.error(`WebSocket error: ${err}`);
471     });
472
473     ws.on("message", (data) => {
474         const response = JSON.parse(data);
475
476         if (response.error) {
477             console.error(`RPC error: ${response.error}`);
478             return;
479         }
480
481         console.log(`RPC response: ${JSON.stringify(response)}`);
482     });
483
484     ws.on("close", () => {
485         console.log("Connection closed");
486     });
487
488     ws.on("error", (err) => {
489         console.error(`WebSocket error: ${err}`);
490     });
491
492     ws.on("message", (data) => {
493         const response = JSON.parse(data);
494
495         if (response.error) {
496             console.error(`RPC error: ${response.error}`);
497             return;
498         }
499
500         console.log(`RPC response: ${JSON.stringify(response)}`);
501     });
502
503     ws.on("close", () => {
504         console.log("Connection closed");
505     });
506
507     ws.on("error", (err) => {
508         console.error(`WebSocket error: ${err}`);
509     });
510
511     ws.on("message", (data) => {
512         const response = JSON.parse(data);
513
514         if (response.error) {
515             console.error(`RPC error: ${response.error}`);
516             return;
517        }
518
519         console.log(`RPC response: ${JSON.stringify(response)}`);
520     });
521
522     ws.on("close", () => {
523         console.log("Connection closed");
524     });
525
526     ws.on("error", (err) => {
527         console.error(`WebSocket error: ${err}`);
528     });
529
530     ws.on("message", (data) => {
531         const response = JSON.parse(data);
532
533         if (response.error) {
534             console.error(`RPC error: ${response.error}`);
535             return;
536        }
537
538         console.log(`RPC response: ${JSON.stringify(response)}`);
539     });
540
541     ws.on("close", () => {
542         console.log("Connection closed");
543     });
544
545     ws.on("error", (err) => {
546         console.error(`WebSocket error: ${err}`);
547     });
548
549     ws.on("message", (data) => {
550         const response = JSON.parse(data);
551
552         if (response.error) {
553             console.error(`RPC error: ${response.error}`);
554             return;
555        }
556
557         console.log(`RPC response: ${JSON.stringify(response)}`);
558     });
559
560     ws.on("close", () => {
561         console.log("Connection closed");
562     });
563
564     ws.on("error", (err) => {
565         console.error(`WebSocket error: ${err}`);
566     });
567
568     ws.on("message", (data) => {
569         const response = JSON.parse(data);
570
571         if (response.error) {
572             console.error(`RPC error: ${response.error}`);
573             return;
574        }
575
576         console.log(`RPC response: ${JSON.stringify(response)}`);
577     });
578
579     ws.on("close", () => {
580         console.log("Connection closed");
581     });
582
583     ws.on("error", (err) => {
584         console.error(`WebSocket error: ${err}`);
585     });
586
587     ws.on("message", (data) => {
588         const response = JSON.parse(data);
589
590         if (response.error) {
591             console.error(`RPC error: ${response.error}`);
592             return;
593        }
594
595         console.log(`RPC response: ${JSON.stringify(response)}`);
596     });
597
598     ws.on("close", () => {
599         console.log("Connection closed");
600     });
601
602     ws.on("error", (err) => {
603         console.error(`WebSocket error: ${err}`);
604     });
605
606     ws.on("message", (data) => {
607         const response = JSON.parse(data);
608
609         if (response.error) {
610             console.error(`RPC error: ${response.error}`);
611             return;
612        }
613
614         console.log(`RPC response: ${JSON.stringify(response)}`);
615     });
616
617     ws.on("close", () => {
618         console.log("Connection closed");
619     });
620
621     ws.on("error", (err) => {
622         console.error(`WebSocket error: ${err}`);
623     });
624
625     ws.on("message", (data) => {
626         const response = JSON.parse(data);
627
628         if (response.error) {
629             console.error(`RPC error: ${response.error}`);
630             return;
631        }
632
633         console.log(`RPC response: ${JSON.stringify(response)}`);
634     });
635
636     ws.on("close", () => {
637         console.log("Connection closed");
638     });
639
640     ws.on("error", (err) => {
641         console.error(`WebSocket error: ${err}`);
642     });
643
644     ws.on("message", (data) => {
645         const response = JSON.parse(data);
646
647         if (response.error) {
648             console.error(`RPC error: ${response.error}`);
649             return;
650        }
651
652         console.log(`RPC response: ${JSON.stringify(response)}`);
653     });
654
655     ws.on("close", () => {
656         console.log("Connection closed");
657     });
658
659     ws.on("error", (err) => {
660         console.error(`WebSocket error: ${err}`);
661     });
662
663     ws.on("message", (data) => {
664         const response = JSON.parse(data);
665
666         if (response.error) {
667             console.error(`RPC error: ${response.error}`);
668             return;
669        }
670
671         console.log(`RPC response: ${JSON.stringify(response)}`);
672     });
673
674     ws.on("close", () => {
675         console.log("Connection closed");
676     });
677
678     ws.on("error", (err) => {
679         console.error(`WebSocket error: ${err}`);
680     });
681
682     ws.on("message", (data) => {
683         const response = JSON.parse(data);
684
685         if (response.error) {
686             console.error(`RPC error: ${response.error}`);
687             return;
688        }
689
690         console.log(`RPC response: ${JSON.stringify(response)}`);
691     });
692
693     ws.on("close", () => {
694         console.log("Connection closed");
695     });
696
697     ws.on("error", (err) => {
698         console.error(`WebSocket error: ${err}`);
699     });
700
701     ws.on("message", (data) => {
702         const response = JSON.parse(data);
703
704         if (response.error) {
705             console.error(`RPC error: ${response.error}`);
706             return;
707        }
708
709         console.log(`RPC response: ${JSON.stringify(response)}`);
710     });
711
712     ws.on("close", () => {
713         console.log("Connection closed");
714     });
715
716     ws.on("error", (err) => {
717         console.error(`WebSocket error: ${err}`);
718     });
719
720     ws.on("message", (data) => {
721         const response = JSON.parse(data);
722
723         if (response.error) {
724             console.error(`RPC error: ${response.error}`);
725             return;
726        }
727
728         console.log(`RPC response: ${JSON.stringify(response)}`);
729     });
730
731     ws.on("close", () => {
732         console.log("Connection closed");
733     });
734
735     ws.on("error", (err) => {
736         console.error(`WebSocket error: ${err}`);
737     });
738
739     ws.on("message", (data) => {
740         const response = JSON.parse(data);
741
742         if (response.error) {
743             console.error(`RPC error: ${response.error}`);
744             return;
745        }
746
747         console.log(`RPC response: ${JSON.stringify(response)}`);
748     });
749
750     ws.on("close", () => {
751         console.log("Connection closed");
752     });
753
754     ws.on("error", (err) => {
755         console.error(`WebSocket error: ${err}`);
756     });
757
758     ws.on("message", (data) => {
759         const response = JSON.parse(data);
760
761         if (response.error) {
762             console.error(`RPC error: ${response.error}`);
763             return;
764        }
765
766         console.log(`RPC response: ${JSON.stringify(response)}`);
767     });
768
769     ws.on("close", () => {
770         console.log("Connection closed");
771     });
772
773     ws.on("error", (err) => {
774         console.error(`WebSocket error: ${err}`);
775     });
776
777     ws.on("message", (data) => {
778         const response = JSON.parse(data);
779
780         if (response.error) {
781             console.error(`RPC error: ${response.error}`);
782             return;
783        }
784
785         console.log(`RPC response: ${JSON.stringify(response)}`);
786     });
787
788     ws.on("close", () => {
789         console.log("Connection closed");
790     });
791
792     ws.on("error", (err) => {
793         console.error(`WebSocket error: ${err}`);
794     });
795
796     ws.on("message", (data) => {
797         const response = JSON.parse(data);
798
799         if (response.error) {
800             console.error(`RPC error: ${response.error}`);
801             return;
802        }
803
804         console.log(`RPC response: ${JSON.stringify(response)}`);
805     });
806
807     ws.on("close", () => {
808         console.log("Connection closed");
809     });
810
811     ws.on("error", (err) => {
812         console.error(`WebSocket error: ${err}`);
813     });
814
815     ws.on("message", (data) => {
816         const response = JSON.parse(data);
817
818         if (response.error) {
819             console.error(`RPC error: ${response.error}`);
820             return;
821        }
822
823         console.log(`RPC response: ${JSON.stringify(response)}`);
824     });
825
826     ws.on("close", () => {
827         console.log("Connection closed");
828     });
829
830     ws.on("error", (err) => {
831         console.error(`WebSocket error: ${err}`);
832     });
833
834     ws.on("message", (data) => {
835         const response = JSON.parse(data);
836
837         if (response.error) {
838             console.error(`RPC error: ${response.error}`);
839             return;
840        }
841
842         console.log(`RPC response: ${JSON.stringify(response)}`);
843     });
844
845     ws.on("close", () => {
846         console.log("Connection closed");
847     });
848
849     ws.on("error", (err) => {
850         console.error(`WebSocket error: ${err}`);
851     });
852
853     ws.on("message", (data) => {
854         const response = JSON.parse(data);
855
856         if (response.error) {
857             console.error(`RPC error: ${response.error}`);
858             return;
859        }
860
861         console.log(`RPC response: ${JSON.stringify(response)}`);
862     });
863
864     ws.on("close", () => {
865         console.log("Connection closed");
866     });
867
868     ws.on("error", (err) => {
869         console.error(`WebSocket error: ${err}`);
870     });
871
872     ws.on("message", (data) => {
873         const response = JSON.parse(data);
874
875         if (response.error) {
876             console.error(`RPC error: ${response.error}`);
877             return;
878        }
879
880         console.log(`RPC response: ${JSON.stringify(response)}`);
881     });
882
883     ws.on("close", () => {
884         console.log("Connection closed");
885     });
886
887     ws.on("error", (err) => {
888         console.error(`WebSocket error: ${err}`);
889     });
890
891     ws.on("message", (data) => {
892         const response = JSON.parse(data);
893
894         if (response.error) {
895             console.error(`RPC error: ${response.error}`);
896             return;
897        }
898
899         console.log(`RPC response: ${JSON.stringify(response)}`);
900     });
901
902     ws.on("close", () => {
903         console.log("Connection closed");
904     });
905
906     ws.on("error", (err) => {
907         console.error(`WebSocket error: ${err}`);
908     });
909
910     ws.on("message", (data) => {
911         const response = JSON.parse(data);
912
913         if (response.error) {
914             console.error(`RPC error: ${response.error}`);
915             return;
916        }
917
918         console.log(`RPC response: ${JSON.stringify(response)}`);
919     });
920
921     ws.on("close", () => {
922         console.log("Connection closed");
923     });
924
925     ws.on("error", (err) => {
926         console.error(`WebSocket error: ${err}`);
927     });
928
929     ws.on("message", (data) => {
930         const response = JSON.parse(data);
931
932         if (response.error) {
933             console.error(`RPC error: ${response.error}`);
934             return;
935        }
936
937         console.log(`RPC response: ${JSON.stringify(response)}`);
938     });
939
940     ws.on("close", () => {
941         console.log("Connection closed");
942     });
943
944     ws.on("error", (err) => {
945         console.error(`WebSocket error: ${err}`);
946     });
947
948     ws.on("message", (data) => {
949         const response = JSON.parse(data);
950
951         if (response.error) {
952             console.error(`RPC error: ${response.error}`);
953             return;
954        }
955
956         console.log(`RPC response: ${JSON.stringify(response)}`);
957     });
958
959     ws.on("close", () => {
960         console.log("Connection closed");
961     });
962
963     ws.on("error", (err) => {
964         console.error(`WebSocket error: ${err}`);
965     });
966
967     ws.on("message", (data) => {
968         const response = JSON.parse(data);
969
970         if (response.error) {
971             console.error(`RPC error: ${response.error}`);
972             return;
973        }
974
975         console.log(`RPC response: ${JSON.stringify(response)}`);
976     });
977
978     ws.on("close", () => {
979         console.log("Connection closed");
980     });
981
982     ws.on("error", (err) => {
983         console.error(`WebSocket error: ${err}`);
984     });
985
986     ws.on("message", (data) => {
987         const response = JSON.parse(data);
988
989         if (response.error) {
990             console.error(`RPC error: ${response.error}`);
991             return;
992        }
993
994         console.log(`RPC response: ${JSON.stringify(response)}`);
995     });
996
997     ws.on("close", () => {
998         console.log("Connection closed");
999     });
999
1000     ws.on("error", (err) => {
1001         console.error(`WebSocket error: ${err}`);
1002     });
1003
1004     ws.on("message", (data) => {
1005         const response = JSON.parse(data);
1006
1007         if (response.error) {
1008             console.error(`RPC error: ${response.error}`);
1009             return;
1010        }
1011
1012         console.log(`RPC response: ${JSON.stringify(response)}`);
1013     });
1014
1015     ws.on("close", () => {
1016         console.log("Connection closed");
1017     });
1018
1019     ws.on("error", (err) => {
1020         console.error(`WebSocket error: ${err}`);
1021     });
1022
1023     ws.on("message", (data) => {
1024         const response = JSON.parse(data);
1025
1026         if (response.error) {
1027             console.error(`RPC error: ${response.error}`);
1028             return;
1029        }
1030
1031         console.log(`RPC response: ${JSON.stringify(response)}`);
1032     });
1033
1034     ws.on("close", () => {
1035         console.log("Connection closed");
1036     });
1037
1038     ws.on("error", (err) => {
1039         console.error(`WebSocket error: ${err}`);
1040     });
1041
1042     ws.on("message", (data) => {
1043         const response = JSON.parse(data);
1044
1045         if (response.error) {
1046             console.error(`RPC error: ${response.error}`);
1047             return;
1048        }
1049
1050         console.log(`RPC response: ${JSON.stringify(response)}`);
1051     });
1052
1053     ws.on("close", () => {
1054         console.log("Connection closed");
1055     });
1056
1057     ws.on("error", (err) => {
1058         console.error(`WebSocket error: ${err}`);
1059     });
1060
1061     ws.on("message", (data) => {
1062         const response = JSON.parse(data);
1063
1064         if (response.error) {
1065             console.error(`RPC error: ${response.error}`);
1066             return;
1067        }
1068
1069         console.log(`RPC response: ${JSON.stringify(response)}`);
1070     });
1071
1072     ws.on("close", () => {
1073         console.log("Connection closed");
1074     });
1075
1076     ws.on("error", (err) => {
1077         console.error(`WebSocket error: ${err}`);
1078     });
1079
1080     ws.on("message", (data) => {
1081         const response = JSON.parse(data);
1082
1083         if (response.error) {
1084             console.error(`RPC error: ${response.error}`);
1085             return;
1086        }
1087
1088         console.log(`RPC response: ${JSON.stringify(response)}`);
1089     });
1090
1091     ws.on("close", () => {
1092         console.log("Connection closed");
1093     });
1094
1095     ws.on("error", (err) => {
1096         console.error(`WebSocket error: ${err}`);
1097     });
1098
1099     ws.on("message", (data) => {
1100         const response = JSON.parse(data);
1101
1102         if (response.error) {
1103             console.error(`RPC error: ${response.error}`);
1104             return;
1105        }
1106
1107         console.log(`RPC response: ${JSON.stringify(response)}`);
1108     });
1109
1110     ws.on("close", () => {
1111         console.log("Connection closed");
1112     });
1113
1114     ws.on("error", (err) => {
1115         console.error(`WebSocket error: ${err}`);
1116     });
1117
1118     ws.on("message", (data) => {
1119         const response = JSON.parse(data);
1120
1121         if (response.error) {
1122             console.error(`RPC error: ${response.error}`);
1123             return;
1124        }
1125
1126         console.log(`RPC response: ${JSON.stringify(response)}`);
1127     });
1128
1129     ws.on("close", () => {
1130         console.log("Connection closed");
1131     });
1132
1133     ws.on("error", (err) => {
1134         console.error(`WebSocket error: ${err}`);
1135     });
1136
1137     ws.on("message", (data) => {
1138         const response = JSON.parse(data);
1139
1140         if (response.error) {
1141             console.error(`RPC error: ${response.error}`);
1142             return;
1143        }
1144
1145         console.log(`RPC response: ${JSON.stringify(response)}`);
1146     });
1147
1148     ws.on("close", () => {
1149         console.log("Connection closed");
1150     });
1151
1152     ws.on("error", (err) => {
1153         console.error(`WebSocket error: ${err}`);
1154     });
1155
1156     ws.on("message", (data) => {
1157         const response = JSON.parse(data);
1158
1159         if (response.error) {
1160             console.error(`RPC error: ${response.error}`);
1161             return;
1162        }
1163
1164         console.log(`RPC response: ${JSON.stringify(response)}`);
1165     });
1166
1167     ws.on("close", () => {
1168         console.log("Connection closed");
1169     });
1170
1171     ws.on("error", (err) => {
1172         console.error(`WebSocket error: ${err}`);
1173     });
1174
1175     ws.on("message", (data) => {
1176         const response = JSON.parse(data);
1177
1178         if (response.error) {
1179             console.error(`RPC error: ${response.error}`);
1180             return;
1181        }
1182
1183         console.log(`RPC response: ${JSON.stringify(response)}`);
1184     });
1185
1186     ws.on("close", () => {
1187         console.log("Connection closed");
1188     });
1189
1190     ws.on("error", (err) => {
1191         console.error(`WebSocket error: ${err}`);
1192     });
1193
1194     ws.on("message", (data) => {
1195         const response = JSON.parse(data);
1196
1197         if (response.error) {
1198             console.error(`RPC error: ${response.error}`);
1199             return;
1200        }
1201
1202         console.log(`RPC response: ${JSON.stringify(response)}`);
1203     });
1204
1205     ws.on("close", () => {
1206         console.log("Connection closed");
1207     });
1208
1209     ws.on("error", (err) => {
1210         console.error(`WebSocket error: ${err}`);
1211     });
1212
1213     ws.on("message", (data) => {
1214         const response = JSON.parse(data);
1215
1216         if (response.error) {
1217             console.error(`RPC error: ${response.error}`);
1218             return;
1219        }
1220
1221         console.log(`RPC response: ${JSON.stringify(response)}`);
1222     });
1223
1224     ws.on("close", () => {
1225         console.log("Connection closed");
1226     });
1227
1228     ws.on("error", (err) => {
1229         console.error(`WebSocket error: ${err}`);
1230     });
1231
1232     ws.on("message", (data) => {
1233         const response = JSON.parse(data);
1234
1235         if (response.error) {
1236             console.error(`RPC error: ${response.error}`);
1237             return;
1238        }
1239
1240         console.log(`RPC response: ${JSON.stringify(response)}`);
1241     });
1242
1243     ws.on("close", () => {
1244         console.log("Connection closed");
1245     });
1246
1247     ws.on("error", (err) => {
1248         console.error(`WebSocket error: ${err}`);
1249     });
1250
1251     ws.on("message", (data) => {
1252         const response = JSON.parse(data);
1253
1254         if (response.error) {
1255             console.error(`RPC error: ${response.error}`);
1256             return;
1257        }
1258
1259         console.log(`RPC response: ${JSON.stringify(response)}`);
1260     });
1261
1262     ws.on("close", () => {
1263         console.log("Connection closed");
1264     });
1265
1266     ws.on("error", (err) => {
1267         console.error(`WebSocket error: ${err}`);
1268     });
1269
1270     ws.on("message", (data) => {
1271         const response = JSON.parse(data);
1272
1273         if (response.error) {
1274             console.error(`RPC error: ${response.error}`);
1275             return;
1276        }
1277
1278         console.log(`RPC response: ${JSON.stringify(response)}`);
1279     });
1280
1281     ws.on("close", () => {
1282         console.log("Connection closed");
1283     });
1284
1285     ws.on("error", (err) => {
1286         console.error(`WebSocket error: ${err}`);
1287     });
1288
1289     ws.on("message", (data) => {
1290         const response = JSON.parse(data);
1291
1292         if (response.error) {
1293             console.error(`RPC error: ${response.error}`);
1294             return;
1295        }
1296
1297         console.log(`RPC response: ${JSON.stringify(response)}`);
1298     });
1299
1300     ws.on("close", () => {
1301         console.log("Connection closed");
1302     });
1303
1304     ws.on("error", (err) => {
1305         console.error(`WebSocket error: ${err}`);
1306     });
1307
1308     ws.on("message", (data) => {
1309         const response = JSON.parse(data);
1310
1311         if (response.error) {
1312             console.error(`RPC error: ${response.error}`);
1313             return;
1314        }
1315
1316         console.log(`RPC response: ${JSON.stringify(response)}`);
1317     });
1318
1319     ws.on("close", () => {
1320         console.log("Connection closed");
1321     });
1322
1323     ws.on("error", (err) => {
1324         console.error(`WebSocket error: ${err}`);
1325     });
1326
1327     ws.on("message", (data) => {
1328         const response = JSON.parse(data);
1329
1330         if (response.error) {
1331             console.error(`RPC error: ${response.error}`);
1332             return;
1333        }
1334
1335         console.log(`RPC response: ${JSON.stringify(response)}`);
1336     });
1337
1338     ws.on("close", () => {
1339         console.log("Connection closed");
1340     });
1341
1342     ws.on("error", (err) => {
1343         console.error(`WebSocket error: ${err}`);
1344     });
1345
1346     ws.on("message", (data) => {
1347         const response = JSON.parse(data);
1348

```

The screenshot shows a dark-themed code editor interface with the following components:

- File Explorer:** On the left, it displays a tree view of files and folders, including "server.js", "yellow.js", "Web3Wallet.js", and "App.js".
- Code Editor:** The main area shows the content of "yellow.js". The code handles WebSocket events for messages, closes, and errors.
- Terminal:** Below the code editor is a terminal window titled "powershell". It shows the command `npm install web3` being run, followed by audit results and a warning about the MongoDB driver.
- Status Bar:** At the bottom, it displays the file path "C:\Users\hp\OneDrive\Desktop\ansh02\bhilare_projects\doctor-ai-anshul-main\utils\yellow.js", line information "Ln 57, Col 1", and encoding "UTF-8".





Project Links & Info

- Deployed Link: doctoraibyanshul.netlify.app
- LinkedIn: linkedin.com/in/anshul-bhilare-251710290/
- Portfolio: anshulbhilareportfolio.netlify.app
- This project was selected at IIT Delhi Hackathon.
- Developed frontend, backend, Web3, Android integration, and APK.