

Python 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      <title>Oppia AI Assistant Prototype</title>
7      <style>
8          :root {
9              --primary-color: #009688;
10             --primary-dark: #00796b;
11             --secondary-color: #3f51b5;
12             --secondary-dark: #303f9f;
13             --bg-color: #f0f2f5;
14             --card-bg: #ffffff;
15             --text-main: #333333;
16             --text-secondary: #666666;
17             --border-color: #e0e0e0;
18             --success-color: #4CAF50;
19             --warning-color: #FF9800;
20             --error-color: #F44336;
21         }
22
23         body {
24             font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Helvetica, Arial, sans-serif;
25             background-color: var(--bg-color);
26             color: var(--text-main);
27             margin: 0;
28             padding: 20px;
29             line-height: 1.6;
30         }
31
32     /* Layout Structure */
33     .main-wrapper {
34         max-width: 900px;
35         margin: 0 auto;
36     }
37
38     header {
39         text-align: center;
40         margin-bottom: 30px;
41     }
42
43     header h2 {
44         color: var(--primary-color);
45         margin: 0;
46         font-weight: 700;
47     }
48
```

```
49     header p {
50         color: var(--text-secondary);
51         margin-top: 5px;
52     }
53
54     .container {
55         display: flex;
56         flex-direction: row;
57         gap: 24px;
58         align-items: flex-start;
59     }
60
61     /* Responsive Stack */
62     @media (max-width: 768px) {
63         .container {
64             flex-direction: column;
65         }
66     }
67
68     /* Panels */
69     .editor-pane, .ai-pane {
70         flex: 1;
71         background-color: var(--card-bg);
72         border: 1px solid var(--border-color);
73         border-radius: 12px;
74         box-shadow: 0 4px 12px rgba(0,0,0,0.05);
75         display: flex;
76         flex-direction: column;
77         overflow: hidden;
78     }
79
80     .pane-header {
81         padding: 15px 20px;
82         border-bottom: 1px solid var(--border-color);
83         background-color: #fafafa;
84     }
85
86     .pane-header h3 {
87         margin: 0;
88         font-size: 1.1rem;
89         color: var(--text-main);
90         display: flex;
91         align-items: center;
92         gap: 8px;
93     }
94
95     .pane-body {
96         padding: 20px;
97         flex: 1;
98     }
99
100    /* Input Area */
```

```
101     textarea {
102         width: 100%;
103         min-height: 150px;
104         padding: 12px;
105         border: 2px solid var(--border-color);
106         border-radius: 8px;
107         font-family: inherit;
108         font-size: 1rem;
109         resize: vertical;
110         transition: border-color 0.2s;
111         box-sizing: border-box;
112         margin-bottom: 15px;
113     }
114
115     textarea:focus {
116         outline: none;
117         border-color: var(--primary-color);
118     }
119
120     /* Controls */
121     .controls {
122         display: flex;
123         gap: 10px;
124         flex-wrap: wrap;
125         margin-bottom: 20px;
126     }
127
128     button {
129         padding: 10px 18px;
130         border: none;
131         border-radius: 6px;
132         cursor: pointer;
133         font-weight: 600;
134         font-size: 0.95rem;
135         color: white;
136         transition: transform 0.1s, opacity 0.2s;
137         display: inline-flex;
138         align-items: center;
139         gap: 6px;
140     }
141
142     button:active { transform: scale(0.98); }
143     button:disabled { opacity: 0.6; cursor: not-allowed; }
144
145     #btnGen { background-color: var(--primary-color); }
146     #btnGen:hover:not(:disabled) { background-color: var(--primary-dar
k); }
147
148     #btnCheck { background-color: var(--secondary-color); }
149     #btnCheck:hover:not(:disabled) { background-color: var(--secondary-
dark); }
150
```

```
151     /* AI Output Styles */
152     .output-group { margin-bottom: 20px; }
153     .output-group h4 {
154         margin-bottom: 10px;
155         color: var(--text-secondary);
156         font-size: 0.9rem;
157         text-transform: uppercase;
158         letter-spacing: 0.5px;
159     }
160
161     .suggestion-item {
162         background: #f8f9fa;
163         padding: 12px;
164         border: 1px solid var(--border-color);
165         margin-bottom: 8px;
166         border-radius: 6px;
167         cursor: pointer;
168         transition: all 0.2s;
169         position: relative;
170     }
171
172     .suggestion-item:hover {
173         background-color: #e0f2f1;
174         border-color: var(--primary-color);
175         transform: translateX(4px);
176     }
177
178     .suggestion-item::after {
179         content: '+';
180         position: absolute;
181         right: 12px;
182         top: 50%;
183         transform: translateY(-50%);
184         color: var(--primary-color);
185         font-weight: bold;
186     }
187
188     /* Quality Badge Styles */
189     .quality-badge {
190         font-weight: bold;
191         padding: 6px 12px;
192         border-radius: 20px;
193         display: inline-block;
194         font-size: 0.85rem;
195     }
196     .good { color: white; background-color: var(--success-color); }
197     .bad { color: white; background-color: var(--error-color); }
198     .neutral { color: white; background-color: var(--warning-color); }
199
200     .quality-card {
201         background: #f8f9fa;
202         border-radius: 8px;
```

```
203         padding: 15px;
204         border-left: 4px solid var(--border-color);
205     }
206
207     /* Animation Classes */
208     .fade-in { animation: fadeIn 0.5s ease-in; }
209     @keyframes fadeIn { from { opacity: 0; transform: translateY(5px); }
209 } to { opacity: 1; transform: translateY(0); } }
210
211     /* Loader */
212     .loader {
213         display: none;
214         text-align: center;
215         padding: 20px;
216         color: var(--text-secondary);
217     }
218     .spinner {
219         border: 3px solid rgba(0,0,0,0.1);
220         border-top: 3px solid var(--primary-color);
221         border-radius: 50%;
222         width: 24px;
223         height: 24px;
224         animation: spin 1s linear infinite;
225         margin: 0 auto 10px auto;
226     }
227     @keyframes spin { 0% { transform: rotate(0deg); } 100% { transform:
rotate(360deg); } }
228
229     /* Toast Notifications */
230     #toast-container {
231         position: fixed;
232         bottom: 20px;
233         right: 20px;
234         z-index: 1000;
235         display: flex;
236         flex-direction: column;
237         gap: 10px;
238     }
239     .toast {
240         background: #333;
241         color: #fff;
242         padding: 12px 24px;
243         border-radius: 8px;
244         box-shadow: 0 4px 12px rgba(0,0,0,0.15);
245         font-size: 0.9rem;
246         animation: slideIn 0.3s ease-out;
247         max-width: 300px;
248     }
249     @keyframes slideIn { from { transform: translateX(100%); opacity:
0; } to { transform: translateX(0); opacity: 1; } }
250
251     /* Empty State */
```

```
252         .empty-state {
253             text-align: center;
254             color: #999;
255             padding: 40px 20px;
256             font-style: italic;
257         }
258
259     </style>
260 </head>
261 <body>
262
263     <div class="main-wrapper">
264         <header>
265             <h2>Oppia Exploration Editor</h2>
266             <p>AI-Powered Content Generation & Evaluation Tool</p>
267         </header>
268
269         <div class="container">
270             <!-- Left: Content Editor -->
271             <div class="editor-pane">
272                 <div class="pane-header">
273                     <h3>📝 State Content</h3>
274                 </div>
275                 <div class="pane-body">
276                     <p style="margin-top:0; font-size: 0.9em; color: #66
6;">Enter the educational text for this state below:</p>
277
278                     <textarea id="contentInput" placeholder="e.g., Photosyn
thesis is a process used by plants and other organisms to convert light energy into
chemical energy..."></textarea>
279
280                 <div class="controls">
281                     <button id="btnGen" onclick="triggerGeneration()">
282                         <span>💡 </span> Generate Suggestions
283                     </button>
284                     <button id="btnCheck" onclick="triggerEvaluation
()">
285                         <span>📊 </span> Check Quality
286                     </button>
287                 </div>
288
289                 <div id="qualityResults"></div>
290             </div>
291         </div>
292
293         <!-- Right: AI Assistant -->
294         <div class="ai-pane">
295             <div class="pane-header">
296                 <h3>🤖 AI Assistant</h3>
297             </div>
298             <div class="pane-body" id="aiOutputPane">
299                 <div class="empty-state" id="initialState">
```

```
300                                     Suggestions and hints will appear here once you gen
erate content.
301                                     </div>
302
303                                     <!-- Loader -->
304                                     <div class="loader" id="aiLoader">
305                                         <div class="spinner"></div>
306                                         <span>Analyzing content...</span>
307                                     </div>
308
309                                     <!-- Output Container -->
310                                     <div id="suggestions" style="display: none;"></div>
311                                     </div>
312                                     </div>
313                                     </div>
314                                     </div>
315
316                                     <!-- Toast Container -->
317                                     <div id="toast-container"></div>
318
319                                     <script>
320                                         // --- Utility Functions ---
321
322                                         // Custom Toast Notification (replaces native alert)
323                                         function showToast(message, type = 'info') {
324                                             const container = document.getElementById('toast-container');
325                                             const toast = document.createElement('div');
326                                             toast.className = 'toast';
327                                             toast.textContent = message;
328
329                                             if(type === 'error') toast.style.borderLeft = '4px solid #F443
6';
330                                             if(type === 'success') toast.style.borderLeft = '4px solid #4CA
F50';
331
332                                             container.appendChild(toast);
333
334                                             // Remove after 3 seconds
335                                             setTimeout(() => {
336                                                 toast.style.opacity = '0';
337                                                 toast.style.transform = 'translateY(10px)';
338                                                 setTimeout(() => container.removeChild(toast), 300);
339                                             }, 3000);
340                                         }
341
342                                         function setLoading(isLoading) {
343                                             const loader = document.getElementById('aiLoader');
344                                             const suggestions = document.getElementById('suggestions');
345                                             const initialState = document.getElementById('initialState');
346                                             const btnGen = document.getElementById('btnGen');
347                                             const btnCheck = document.getElementById('btnCheck');
```

```

349     if (isLoading) {
350         loader.style.display = 'block';
351         suggestions.style.display = 'none';
352         initialState.style.display = 'none';
353         btnGen.disabled = true;
354         btnCheck.disabled = true;
355     } else {
356         loader.style.display = 'none';
357         suggestions.style.display = 'block';
358         btnGen.disabled = false;
359         btnCheck.disabled = false;
360     }
361 }
362
363 // --- Simulated AI Logic (Backend Replacement) ---
364
365 function getKeywords(text) {
366     const words = text.split(/\s+/).filter(w => w.length > 4);
367     const uniqueWords = [...new Set(words)];
368     // Return top 3 longest words as "topics"
369     return uniqueWords.sort((a,b) => b.length - a.length).slice(0,
3);
370 }
371
372 function simulateAIGeneration(content) {
373     return new Promise((resolve) => {
374         setTimeout(() => {
375             const topics = getKeywords(content);
376             const mainTopic = topics[0] || "this concept";
377
378             const data = {
379                 questions: [
380                     `What are the main characteristics of ${mainTopic}`,
381                     `How does ${mainTopic} affect the surrounding environment?`,
382                     `Can you explain the process of ${mainTopic} in your own words?`,
383                 ],
384                 hint: "Think about the definitions and the key examples provided in the text above.",
385                 feedback: "Not quite! Remember to focus on the specific mechanism mentioned in the second paragraph."
386             };
387             resolve(data);
388         }, 1200); // Simulate network delay
389     });
390 }
391
392 function simulateAIEvaluation(content) {
393     return new Promise((resolve) => {
394         setTimeout(() => {

```

```
395         const wordCount = content.split(/\s+/).length;
396         let score, status, warning, badgeClass;
397
398         // Simple logic for demonstration
399         if (wordCount < 10) {
400             score = 1;
401             status = "Too Short";
402             badgeClass = "bad";
403             warning = "Content is too brief to be educational";
404         } else if (wordCount > 100) {
405             score = 12; // High grade level
406             status = "Too Complex";
407             badgeClass = "bad";
408             warning = "Consider breaking this into smaller chunks.";
409         } else {
410             score = 5 + Math.floor(Math.random() * 3); // Random grade between 5 and 7
411             status = "Good";
412             badgeClass = "good";
413             warning = null;
414         }
415
416         const data = {
417             readability_score: score,
418             status: status,
419             warning: warning,
420             badgeClass: badgeClass
421         };
422         resolve(data);
423     }, 1000);
424 );
425
426
427 // --- Main Interaction Functions ---
428
429 async function triggerGeneration() {
430     const content = document.getElementById('contentInput').value.trim();
431
432     if (!content) {
433         showToast("Please enter some text first.", "error");
434         document.getElementById('contentInput').focus();
435         return;
436     }
437
438     setLoading(true);
439
440     try {
441         // In the real app, this would be: const response = await fetch('/api/generate', ...);
442     } catch (error) {
443         console.error(error);
444         setLoading(false);
445         showToast("An error occurred while generating the report.", "error");
446     }
447 }
```

```

442         const data = await simulateAIGeneration(content);
443         displaySuggestions(data);
444         showToast("Suggestions generated successfully!", "success");
445     } catch (error) {
446         console.error(error);
447         showToast("Failed to generate suggestions.", "error");
448     } finally {
449         setLoading(false);
450     }
451 }
452
453 async function triggerEvaluation() {
454     const content = document.getElementById('contentInput').value.trim();
455
456     if (!content) {
457         showToast("Please enter some text first.", "error");
458         return;
459     }
460
461     // Show a mini loading state specifically for quality if needed,
462     // but we reuse the main loader for this prototype
463     const container = document.getElementById('qualityResults');
464     container.innerHTML = `<div style="padding:10px; color:#666;">Analyzing quality...</div>`;
465
466     try {
467         const data = await simulateAIEvaluation(content);
468         displayQuality(data);
469         showToast("Quality check complete.", "success");
470     } catch (error) {
471         console.error(error);
472         showToast("Failed to evaluate quality.", "error");
473         container.innerHTML = '';
474     }
475 }
476
477 // --- Display Functions ---
478
479 function displaySuggestions(data) {
480     const container = document.getElementById('suggestions');
481     let html = '<div class="fade-in">';
482
483     html += '<div class="output-group">';
484     html += '<h4>Suggested Questions</h4>';
485     if (data.questions && data.questions.length > 0) {
486         data.questions.forEach(q => {
487             const safeQ = q.replace(/\-/g, "&#39;");
488             html += `<div class="suggestion-item" onclick="useSuggestion('${safeQ}')">${q}</div>`;
489         });
490     }
491     container.innerHTML = html;
492 }

```

```

489             });
490         } else {
491             html += '<p>No questions generated. Try using more descriptive text.</p>';
492         }
493         html += '</div>';
494
495         html += '<div class="output-group">';
496         html += '<h4>Suggested Hint</h4>';
497         html += `<div class="suggestion-item" style="cursor:default; border-left: 4px solid #FF9800;">${data_hint}</div>`;
498         html += `</div>`;
499
500         html += '<div class="output-group">';
501         html += '<h4>Default Feedback</h4>';
502         html += `<div class="suggestion-item" style="cursor:default; border-left: 4px solid #2196F3;">${data_feedback}</div>`;
503         html += `</div>`;
504
505         html += `</div>`;
506         container.innerHTML = html;
507     }
508
509     function displayQuality(data) {
510         const container = document.getElementById('qualityResults');
511
512         container.innerHTML =
513             `<div class="fade-in">
514                 <h4>Quality Check Results</h4>
515                 <div class="quality-card" style="border-left-color: ${data.badgeClass === 'good' ? '#4CAF50' : (data.badgeClass === 'bad' ? '#F44336' : '#FF9800')}>
516                     <p style="margin: 0 0 10px 0;"><strong>Readability Level:</strong> Grade ${data.readability_score}</p>
517                     <span class="quality-badge ${data.badgeClass}">${data.status}</span>
518                     ${data.warning ? `<p style="color: #d32f2f; font-size: 0.9em; margin-top: 12px; margin-bottom:0;">⚠️ ${data.warning}</p>` : ''}
519                 </div>
520             </div>
521         `;
522     }
523
524     function useSuggestion(text) {
525         // In a real integration, this might copy to clipboard or fill a form field.
526         // For this UI demo, we show a nice visual feedback.
527         showToast(`Selected: "${text.substring(0, 25)}..."`, "success");
528
529         // Optional: Paste into textarea to show effect
530         // document.getElementById('contentInput').value += `\n\nQ: ${t

```

```
ext}`;  
531         }  
532     </script>  
533 </body>  
534 </html>
```