

■ Hackathon Project - Frontend Code Export

■ Frontend JavaScript/JSX Files:

```
[JS] eslint.config.js
[JS] postcss.config.js
[JS] server\server.js
[JSX] src\App.jsx
[JSX] src\components\blog-generator\BlogGenerator.jsx
[JSX] src\components\blog-generator\BlogInput.jsx
[JSX] src\components\blog-generator\BlogOutput.jsx
[JSX] src\components\blog-generator\ExamplePrompts.jsx
[JSX] src\components\blog-generator\FeatureCards.jsx
[JSX] src\components\ui\alert.jsx
[JSX] src\components\ui\badge.jsx
[JSX] src\components\ui\button.jsx
[JSX] src\components\ui\card.jsx
[JSX] src\components\ui\separator.jsx
[JSX] src\components\ui\tabs.jsx
[JSX] src\components\ui\textarea.jsx
[JS] src\hooks\useBlogGenerator.js
[JS] src\hooks\useLocalStorage.js
[JSX] src\main.jsx
[JS] src\services\blogApi.js
[JS] src\utils\blogMetrics.js
[JS] src\utils\constants.js
[JS] tailwind.config.js
[JS] vite.config.js
```

■ File: eslint.config.js

```
=====
1: import js from '@eslint/js'
2: import globals from 'globals'
3: import reactHooks from 'eslint-plugin-react-hooks'
4: import reactRefresh from 'eslint-plugin-react-refresh'
5: import { defineConfig, globalIgnores } from 'eslint/config'
6:
7: export default defineConfig([
8:   globalIgnores(['dist']),
9:   {
10:    files: ['**/*.js', '**/*.jsx'],
11:    extends: [
12:      js.configs.recommended,
13:      reactHooks.configs.flat.recommended,
14:      reactRefresh.configs.vite,
15:    ],
16:    languageOptions: {
17:      ecmaVersion: 2020,
18:      globals: globals.browser,
19:      parserOptions: {
20:        ecmaVersion: 'latest',
21:        ecmaFeatures: { jsx: true },
22:        sourceType: 'module',
23:      },
24:    },
25:    rules: {
26:      'no-unused-vars': ['error', { varsIgnorePattern: '^_[A-Z_]' }],
27:    },
28:  },
29: ])
```

■ File: postcss.config.js

```
=====
1: export default {
2:   plugins: {
3:     tailwindcss: {},
4:     autoprefixer: {},
5:   },
6: };
```

■ File: server/server.js

```
=====
1: import express from "express";
2: import cors from "cors";
3: import dotenv from "dotenv";
4: import fetch from "node-fetch";
5:
6: dotenv.config();
7:
8: const app = express();
9: app.use(cors());
10: app.use(express.json());
11:
12: app.post("/api/generate-blog", async (req, res) => {
13:   try {
14:     const { prompt } = req.body;
15:
16:     if (!prompt || prompt.trim() === "") {
17:       return res.status(400).json({ error: "Prompt is required" });
18:     }
19:
20:     const groqResponse = await fetch(
21:       "https://api.groq.com/openai/v1/chat/completions",
22:       {
23:         method: "POST",
24:         headers: {
25:           "Authorization": `Bearer ${process.env.GROQ_API_KEY}`,
26:           "Content-Type": "application/json",
```

```

27:         },
28:         body: JSON.stringify({
29:             model: "llama-3.3-70b-versatile",
30:             messages: [
31:                 {
32:                     role: "user",
33:                     content: `
34: You are a professional blog writer. Write a polished, structured, 1000-word blog in clean Markdown that
35:
36: Your response MUST follow this exact structure with correct spacing:
37:
38: # Title of the Blog
39:
40: A clear, engaging introduction of 3-5 sentences explaining the topic and its importance.
41:
42: ---
43:
44: ## 1. Introduction
45:
46: Write 1-2 detailed paragraphs covering background, significance, and relevance.
47:
48: ---
49:
50: ## 2. Key Applications
51:
52: ### a. Subheading Title
53: Short explanation paragraph.
54:
55: ### b. Subheading Title
56: Short explanation paragraph.
57:
58: ### c. Subheading Title
59: Short explanation paragraph.
60:
61: (Add more subsections only if needed.)
62:
63: ---
64:
65: ## 3. Benefits
66:
67: Write this as a proper Markdown bullet list.
68: You MUST follow the format below EXACTLY:
69:
70: Example (DO NOT include these exact words):
71: - This is an example of a correct bullet format.
72: - Every bullet MUST start with a hyphen "-".
73: - Never merge bullets into one paragraph.
74:
75: Now write the actual benefits below, following that format:
76:
77: - Benefit 1 explanation sentence.
78: - Benefit 2 explanation sentence.
79: - Benefit 3 explanation sentence.
80: - Benefit 4 explanation sentence.
81:
82: ---
83:
84: ## 4. Challenges
85:
86: Write this section as a proper Markdown bullet list.
87: Follow the same formatting rules:
88:
89: Example (DO NOT include these exact words):
90: - This is an example challenge bullet.
91: - Each bullet MUST start with a hyphen "-".
92: - Never place multiple bullets in a single line.
93: - Never write bullets as paragraphs.
94:
95: Now write the actual challenges below:
96:
97: - Challenge 1 explanation sentence.
98: - Challenge 2 explanation sentence.
99: - Challenge 3 explanation sentence.

```

```

100: - Challenge 4 explanation sentence.
101:
102: ---
103:
104: ## **5. Future Scope**
105:
106: Write 1?2 paragraphs describing future possibilities, innovations, and long-term impact.
107:
108: ---
109:
110: ## **6. Conclusion**
111:
112: Write a final paragraph summarizing the topic clearly and professionally.
113:
114: ---
115:
116: STRICT RULES:
117: - ALWAYS add blank lines between headings, paragraphs, and bullets.
118: - NEVER use "?" bullets. ONLY use "-" for lists.
119: - NEVER merge multiple bullet points into one paragraph.
120: - NEVER use HTML.
121: - Output ONLY the blog. No additional explanation.
122: - Title, headings, and subheadings MUST match this Markdown structure.
123:
124: Write the complete blog on the topic: ${prompt}
125: `
126:         },
127:       ],
128:       temperature: 0.7,
129:       max_completion_tokens: 2000,
130:     })),
131:   }
132: );
133:
134: const data = await groqResponse.json();
135:
136: if (!groqResponse.ok) {
137:   return res.status(500).json({
138:     error: data.error?.message || "Groq API failed",
139:   });
140: }
141:
142: const blogText = data.choices?.[0]?.message?.content || "";
143:
144: if (!blogText.trim()) {
145:   return res.status(500).json({ error: "Empty blog response" });
146: }
147:
148: return res.json({ blog: blogText });
149:
150: } catch (error) {
151:   console.error("SERVER ERROR:", error);
152:   return res.status(500).json({ error: "Server error" });
153: }
154: });
155:
156: app.listen(process.env.PORT, () => {
157:   console.log(`Server running on port ${process.env.PORT}`);
158: });

```

■ File: src\App.jsx

```

=====
1: // src/App.jsx
2: import { Toaster } from 'sonner';
3: import BlogGenerator from '../components/blog-generator/BlogGenerator';
4: import './App.css';
5:
6: function App() {
7:   return (
8:     <>
9:       <BlogGenerator />

```

```

10:     <Toaster
11:       position="top-right"
12:       richColors
13:       closeButton
14:       duration={4000}
15:     />
16:   </>
17: );
18: }
19:
20: export default App;

```

■ File: src\components\blog-generator\BlogGenerator.jsx

```

1: // src/components/blog-generator/BlogGenerator.jsx
2: import { useEffect } from "react";
3: import { toast } from "sonner";
4: import { useBlogGenerator } from "../../hooks/useBlogGenerator";
5: import BlogInput from "../BlogInput";
6: import BlogOutput from "../BlogOutput";
7:
8: export default function BlogGenerator() {
9:   const {
10:     prompt,
11:     setPrompt,
12:     blog,
13:     isLoading,
14:     error,
15:     generate,
16:     reset,
17:     copyToClipboard,
18:   } = useBlogGenerator();
19:
20:   const handleGenerate = async () => {
21:     const result = await generate();
22:
23:     if (result.success) {
24:       toast.success("Blog generated successfully!", {
25:         description: `${result.blog.metrics.wordCount} words in ${result.blog.metrics.generationTime}s`,
26:       });
27:
28:       // Scroll to result
29:       setTimeout(() => {
30:         document.getElementById("blog-output")?.scrollIntoView({
31:           behavior: "smooth",
32:           block: "start",
33:         });
34:       }, 100);
35:     } else {
36:       toast.error("Generation failed", {
37:         description: result.error,
38:       });
39:     }
40:   };
41:
42:   const handleCopy = async () => {
43:     const result = await copyToClipboard();
44:     if (result.success) {
45:       toast.success("Copied to clipboard!");
46:     } else {
47:       toast.error("Failed to copy to clipboard");
48:     }
49:     return result;
50:   };
51:
52:   const handleReset = () => {
53:     reset();
54:     toast.info("Generator reset");
55:   };
56:
57:   const handleExampleSelect = (example) => {

```

```

58:     setPrompt(example);
59:     toast.info("Example prompt loaded");
60:   };
61:
62:   return (
63:     <div>
64:       <div className="container mx-auto px-4 py-8 md:py-12">
65:         <div className="max-w-7xl mx-auto space-y-8">
66:           <BlogInput
67:             prompt={prompt}
68:             setPrompt={setPrompt}
69:             isLoading={isLoading}
70:             error={error}
71:             onGenerate={handleGenerate}
72:             onReset={handleReset}
73:             onExampleSelect={handleExampleSelect}
74:           />
75:
76:           {blog && <BlogOutput blog={blog} onCopy={handleCopy} />}
77:         </div>
78:       </div>
79:     </div>
80:   );
81: }

```

File: src/components/blog-generator/BlogInput.jsx

```

=====
1: // src/components/blog-generator/BlogInput.jsx
2: import {
3:   Loader2,
4:   Sparkles,
5:   Zap,
6:   ArrowRight,
7:   RefreshCw,
8:   Brain,
9:   AlertCircle,
10: } from "lucide-react";
11: import {
12:   Card,
13:   CardContent,
14:   CardDescription,
15:   CardHeader,
16:   CardTitle,
17: } from "../ui/card";
18: import { Button } from "../ui/button";
19: import { Textarea } from "../ui/textarea";
20: import { Alert, AlertDescription } from "../ui/alert";
21: import ExamplePrompts from "../ExamplePrompts";
22: import { PROMPT_CONFIG } from "../../utils/constants";
23:
24: export default function BlogInput({
25:   prompt,
26:   setPrompt,
27:   isLoading,
28:   error,
29:   onGenerate,
30:   onReset,
31: }) {
32:   const handleExampleSelect = (example) => {
33:     setPrompt(example);
34:   };
35:
36:   return (
37:     <Card className="border-2 shadow-lg">
38:       <CardHeader>
39:         <CardTitle className="flex items-center gap-2 text-2xl">
40:           <Sparkles className="h-6 w-6 text-blue-600" />
41:           Create Your Blog
42:         </CardTitle>
43:         <CardDescription className="text-base">
44:           Describe your topic in detail. The more specific you are, the better

```

```

45:         the results.
46:     </CardDescription>
47: </CardHeader>
48: <CardContent className="space-y-6">
49:     <div className="space-y-3">
50:         <Textarea
51:             placeholder={PROMPT_CONFIG.PLACEHOLDER}
52:             value={prompt}
53:             onChange={(e) => setPrompt(e.target.value)}
54:             rows={6}
55:             className="resize-none text-base"
56:             disabled={isLoading}
57:             maxLength={PROMPT_CONFIG.MAX_LENGTH}
58:         />
59:         <div className="flex items-center justify-between text-sm text-muted-foreground">
60:             <span>
61:                 {prompt.length}/{PROMPT_CONFIG.MAX_LENGTH} characters
62:             </span>
63:             {prompt && (
64:                 <Button
65:                     variant="ghost"
66:                     size="sm"
67:                     onClick={onReset}
68:                     className="h-8"
69:                     disabled={isLoading}
70:                 >
71:                     <RefreshCw className="h-3 w-3 mr-1" />
72:                     Clear
73:                 </Button>
74:             )}
75:         </div>
76:     </div>
77:
78:     <ExamplePrompts
79:         onSelectPrompt={handleExampleSelect}
80:         disabled={isLoading}
81:     />
82:
83:     {error && (
84:         <Alert variant="destructive">
85:             <AlertCircle className="h-4 w-4" />
86:             <AlertDescription>{error}</AlertDescription>
87:         </Alert>
88:     )}
89:
90:     <Button
91:         onClick={onGenerate}
92:         disabled={isLoading || !prompt.trim()}
93:         className="w-half justify-center h-12 text-base"
94:         size="lg"
95:     >
96:         {isLoading ? (
97:             <>
98:                 <Loader2 className="mr-2 h-5 w-5 animate-spin" />
99:                 Generating Your Blog...
100:             </>
101:         ) : (
102:             <>
103:                 <Zap className="mr-2 h-5 w-5" />
104:                 Generate Professional Blog
105:             </>
106:         )}
107:     </Button>
108: </CardContent>
109: </Card>
110: );
111: }

```

■ File: src\components\blog-generator\BlogOutput.jsx

```
=====
1: // src/components/blog-generator/BlogOutput.jsx
2: import { useState } from "react";
3: import { Copy, Download, CheckCircle } from "lucide-react";
4: import ReactMarkdown from "react-markdown";
5: import {
6:   Card,
7:   CardContent,
8:   CardDescription,
9:   CardHeader,
10:  CardTitle,
11: } from "../ui/card";
12: import { Button } from "../ui/button";
13: import { Separator } from "../ui/separator";
14: import { Tabs, TabsContent, TabsList, TabsTrigger } from "../ui/tabs";
15: import { downloadBlog } from "../../utils/blogMetrics";
16:
17: export default function BlogOutput({ blog, onCopy }) {
18:   const [viewMode, setViewMode] = useState("preview");
19:   const [copied, setCopied] = useState(false);
20:
21:   const handleCopy = async () => {
22:     const result = await onCopy();
23:     if (result.success) {
24:       setCopied(true);
25:       setTimeout(() => setCopied(false), 2000);
26:     }
27:   };
28:
29:   const handleDownload = () => {
30:     downloadBlog(blog.content);
31:   };
32:
33:   return (
34:     <Card className="border-2 shadow-lg" id="blog-output">
35:       <CardHeader>
36:         <div className="flex flex-col sm:flex-row sm:items-center justify-between gap-4">
37:           <div className="space-y-1">
38:             <CardTitle className="text-2xl">Generated Blog</CardTitle>
39:             <CardDescription>
40:               Created on {new Date(blog.timestamp).toLocaleString()}
41:             </CardDescription>
42:           </div>
43:
44:           <div className="flex gap-2">
45:             <Button
46:               variant="outline"
47:               size="sm"
48:               onClick={handleCopy}
49:               className="gap-2"
50:             >
51:               {copied ? (
52:                 <>
53:                   <CheckCircle className="h-4 w-4 text-green-600" />
54:                   Copied!
55:                 </>
56:               ) : (
57:                 <>
58:                   <Copy className="h-4 w-4" />
59:                   Copy
60:                 </>
61:               )}
62:             </Button>
63:
64:             <Button
65:               variant="outline"
66:               size="sm"
67:               onClick={handleDownload}
68:               className="gap-2"
69:             >
70:               <Download className="h-4 w-4" />
71:               Download

```



```

72:         </Button>
73:     </div>
74: </div>
75:
76: <Separator className="mt-4" />
77:
78: { /* Metrics */}
79: <div className="grid grid-cols-2 sm:grid-cols-4 gap-4 pt-4">
80:     <div className="space-y-1">
81:         <p className="text-sm text-muted-foreground">Words</p>
82:         <p className="text-2xl font-bold">
83:             {blog.metrics.wordCount.toLocaleString()}
84:         </p>
85:     </div>
86:     <div className="space-y-1">
87:         <p className="text-sm text-muted-foreground">Characters</p>
88:         <p className="text-2xl font-bold">
89:             {blog.metrics.characterCount.toLocaleString()}
90:         </p>
91:     </div>
92:     <div className="space-y-1">
93:         <p className="text-sm text-muted-foreground">Read Time</p>
94:         <p className="text-2xl font-bold">{blog.metrics.readingTime} min</p>
95:     </div>
96:     <div className="space-y-1">
97:         <p className="text-sm text-muted-foreground">Generated</p>
98:         <p className="text-2xl font-bold">{blog.metrics.generationTime}s</p>
99:     </div>
100: </div>
101: </CardHeader>
102:
103: <CardContent>
104:     <Tabs value={viewMode} onValueChange={setViewMode}>
105:         <TabList className="grid w-full max-w-md grid-cols-2">
106:             <TabTrigger value="preview">Preview</TabTrigger>
107:             <TabTrigger value="markdown">Markdown</TabTrigger>
108:         </TabList>
109:
110:         { /* ----- PREVIEW MODE ----- */}
111:         <TabsContent value="preview" className="mt-6">
112:             <div
113:                 className="
114:                 prose prose-slate dark:prose-invert
115:                 text-left max-w-none
116:                 prose-headings:font-bold
117:                 prose-h1:text-3xl
118:                 prose-h2:text-2xl
119:                 prose-h3:text-xl
120:                 prose-p:leading-7
121:                 prose-li:my-1
122:
123:                 /* ---- FORCE BULLET POINTS ---- */
124:                 prose-ul:list-disc
125:                 prose-ol:list-decimal
126:                 prose-li:marker:text-current
127:                 prose-ul:pl-6
128:                 prose-ol:pl-6
129:
130:                 /* extra enforcement (cannot be overridden) */
131:                 [&_ul]:list-disc
132:                 [&_li]:list-disc
133:             ">
134:                 >
135:                 <ReactMarkdown>{blog.content}</ReactMarkdown>
136:             </div>
137:         </TabsContent>
138:
139:         { /* ----- MARKDOWN MODE ----- */}
140:         <TabsContent value="markdown" className="mt-6">
141:             <pre className="p-6 bg-slate-50 dark:bg-slate-900 rounded-lg overflow-x-auto text-sm">
142:                 <code>{blog.content}</code>
143:             </pre>
144:         </TabsContent>

```

```

145:         </Tabs>
146:     </CardContent>
147: </Card>
148: );
149: }

```

■ File: src\components\blog-generator\ExamplePrompts.jsx

```

1: // src/components/blog-generator/ExamplePrompts.jsx
2: import { FileText } from "lucide-react";
3: import { EXAMPLE_PROMPTS } from "../../utils/constants";
4:
5: export default function ExamplePrompts({ onSelectPrompt, disabled }) {
6:     return (
7:         <div className="space-y-3">
8:             <p className="text-sm font-medium">
9:                 Try these examples
10:             </p>
11:             <div className="grid gap-2 sm:grid-cols-2">
12:                 {EXAMPLE_PROMPTS.map((example, index) => (
13:                     <button
14:                         key={index}
15:                         onClick={() => onSelectPrompt(example)}
16:                         disabled={disabled}
17:                         className="text-left p-3 rounded-lg border border-dashed hover:border-solid hover:bg-accent tra
18:                     >
19:                         <span className="flex items-center gap-2">
20:                             <FileText className="h-4 w-4 shrink-0 text-muted-foreground" />
21:                             <span className="truncate">{example}</span>
22:                         </span>
23:                     </button>
24:                 ))}
25:             </div>
26:         </div>
27:     );
28: }

```

■ File: src\components\blog-generator\FeatureCards.jsx

```

1: // src/components/blog-generator/FeatureCards.jsx
2: import { FileText, Sparkles, Zap, CheckCircle } from "lucide-react";
3: import { Card, CardContent } from "../ui/card";
4:
5: const iconMap = {
6:     FileText,
7:     Sparkles,
8:     Zap,
9:     CheckCircle,
10: };
11:
12: export default function FeatureCards({ features }) {
13:     return (
14:         <div className="grid gap-6 sm:grid-cols-2 lg:grid-cols-4">
15:             {features.map((feature, index) => {
16:                 const Icon = iconMap[feature.icon];
17:                 return (
18:                     <Card key={index} className="border-2">
19:                         <CardContent className="pt-6">
20:                             <div className="space-y-3">
21:                                 <div className="flex h-12 w-12 items-center justify-center rounded-lg bg-blue-100 dark:bg-b
22:                                 <Icon className="h-6 w-6 text-blue-600 dark:text-blue-400" />
23:                             </div>
24:                             <h3 className="font-semibold text-lg">{feature.title}</h3>
25:                             <p className="text-sm text-muted-foreground">
26:                                 {feature.description}
27:                             </p>
28:                         </div>
29:                     </CardContent>
30:                 </Card>

```

```

31:         );
32:     }}}}
33: </div>
34: );
35: }

```

■ File: src\components\ui\alert.jsx

```

=====
1: // src/components/ui/alert.jsx
2: import * as React from "react"
3:
4: const alertVariants = {
5:   default: "bg-background text-foreground",
6:   destructive: "border-destructive/50 text-destructive dark:border-destructive [&svg]:text-destructive",
7: }
8:
9: const Alert = React.forwardRef(({ className = "", variant = "default", ...props }, ref) => {
10:   const variantClass = alertVariants[variant] || alertVariants.default
11:
12:   return (
13:     <div
14:       ref={ref}
15:       role="alert"
16:       className={`relative w-full rounded-lg border p-4 [&svg~*]:pl-7 [&svg+div]:translate-y-[-3px] [&svg]:h-5`
17:       {...props}
18:     />
19:   )
20: })
21: Alert.displayName = "Alert"
22:
23: const AlertTitle = React.forwardRef(({ className = "", ...props }, ref) => (
24:   <h5
25:     ref={ref}
26:     className={`mb-1 font-medium leading-none tracking-tight ${className}`}
27:     {...props}
28:   />
29: ))
30: AlertTitle.displayName = "AlertTitle"
31:
32: const AlertDescription = React.forwardRef(({ className = "", ...props }, ref) => (
33:   <div
34:     ref={ref}
35:     className={`text-sm [&p]:leading-relaxed ${className}`}
36:     {...props}
37:   />
38: ))
39: AlertDescription.displayName = "AlertDescription"
40:
41: export { Alert, AlertTitle, AlertDescription }
=====

```

■ File: src\components\ui\badge.jsx

```

=====
1: // src/components/ui/badge.jsx
2: import * as React from "react";
3:
4: const badgeVariants = {
5:   default:
6:     "border-transparent bg-primary text-primary-foreground hover:bg-primary/80",
7:   secondary:
8:     "border-transparent bg-secondary text-secondary-foreground hover:bg-secondary/80",
9:   destructive:
10:    "border-transparent bg-destructive text-destructive-foreground hover:bg-destructive/80",
11:   outline: "text-foreground",
12: };
13:
14: function Badge({ className = "", variant = "default", ...props }) {
15:   const variantClass = badgeVariants[variant] || badgeVariants.default;
16:
17:   return (

```

```

18:     <div
19:       className={`inline-flex items-center rounded-full border px-2.5 py-0.5 text-xs font-semibold transition
20:       {...props}
21:     />
22:   );
23: }
24:
25: export { Badge };

```

■ File: src\components\ui\button.jsx

```

=====
1: // src/components/ui/button.jsx
2: import * as React from "react";
3:
4: const buttonVariants = {
5:   default: "bg-primary text-primary-foreground hover:bg-primary/90",
6:   destructive:
7:     "bg-destructive text-destructive-foreground hover:bg-destructive/90",
8:   outline:
9:     "border border-input bg-background hover:bg-accent hover:text-accent-foreground",
10:  secondary: "bg-secondary text-secondary-foreground hover:bg-secondary/80",
11:  ghost: "hover:bg-accent hover:text-accent-foreground",
12:  link: "text-primary underline-offset-4 hover:underline",
13: };
14:
15: const buttonSizes = {
16:   default: "h-10 px-4 py-2",
17:   sm: "h-9 rounded-md px-3",
18:   lg: "h-11 rounded-md px-8",
19:   icon: "h-10 w-10",
20: };
21:
22: const Button = React.forwardRef(
23:   (
24:     { className = "", variant = "default", size = "default", ...props },
25:     ref
26:   ) => {
27:     const variantClass = buttonVariants[variant] || buttonVariants.default;
28:     const sizeClass = buttonSizes[size] || buttonSizes.default;
29:
30:     return (
31:       <button
32:         className={`inline-flex items-center justify-center whitespace-nowrap rounded-md text-sm font-medium
33:         ref={ref}
34:         {...props}
35:       />
36:     );
37:   }
38: );
39: Button.displayName = "Button";
40:
41: export { Button };

```

■ File: src\components\ui\card.jsx

```

=====
1: // src/components/ui/card.jsx
2: import * as React from "react";
3:
4: const Card = React.forwardRef(({ className = "", ...props }, ref) => (
5:   <div
6:     ref={ref}
7:     className={`rounded-lg border bg-card text-card-foreground shadow-sm ${className}`}
8:     {...props}
9:   />
10: ));
11: Card.displayName = "Card";
12:
13: const CardHeader = React.forwardRef(({ className = "", ...props }, ref) => (
14:   <div

```

```

15:     ref={ref}
16:     className={`flex flex-col space-y-1.5 p-6 ${className}`}
17:     {...props}
18:   />
19: ));
20: CardHeader.displayName = "CardHeader";
21:
22: const CardTitle = React.forwardRef(({ className = "", ...props }, ref) => (
23:   <h3
24:     ref={ref}
25:     className={`text-2xl font-semibold leading-none tracking-tight ${className}`}
26:     {...props}
27:   />
28: ));
29: CardTitle.displayName = "CardTitle";
30:
31: const CardDescription = React.forwardRef(
32:   ({ className = "", ...props }, ref) => (
33:     <p
34:       ref={ref}
35:       className={`text-sm text-muted-foreground ${className}`}
36:       {...props}
37:     />
38:   )
39: );
40: CardDescription.displayName = "CardDescription";
41:
42: const CardContent = React.forwardRef(({ className = "", ...props }, ref) => (
43:   <div ref={ref} className={`p-6 pt-0 ${className}`} {...props} />
44: ));
45: CardContent.displayName = "CardContent";
46:
47: const CardFooter = React.forwardRef(({ className = "", ...props }, ref) => (
48:   <div
49:     ref={ref}
50:     className={`flex items-center p-6 pt-0 ${className}`}
51:     {...props}
52:   />
53: ));
54: CardFooter.displayName = "CardFooter";
55:
56: export {
57:   Card,
58:   CardHeader,
59:   CardFooter,
60:   CardTitle,
61:   CardDescription,
62:   CardContent,
63: };

```

■ File: src/components/ui/separator.jsx

```

1: // src/components/ui/separator.jsx
2: import * as React from "react";
3:
4: const Separator = React.forwardRef(
5:   (
6:     { className = "", orientation = "horizontal", decorative = true, ...props },
7:     ref
8:   ) => (
9:     <div
10:       ref={ref}
11:       role={decorative ? "none" : "separator"}
12:       aria-orientation={orientation}
13:       className={`shrink-0 bg-border ${
14:         orientation === "horizontal" ? "h-[1px] w-full" : "h-full w-[1px]"
15:       } ${className}`}
16:       {...props}
17:     />
18:   )
19: );

```

```

20: Separator.displayName = "Separator";
21:
22: export { Separator };

```

■ File: src\components\ui\tabs.jsx

```

1: // src/components/ui/tabs.jsx
2: import * as React from "react";
3:
4: const TabsContext = React.createContext({});
5:
6: const Tabs = ({
7:   value,
8:   onValueChange,
9:   defaultValue,
10:  children,
11:  className = "",
12: }) => {
13:   const [selectedValue, setSelectedValue] = React.useState(
14:     defaultValue || value
15:   );
16:
17:   React.useEffect(() => {
18:     if (value !== undefined) {
19:       setSelectedValue(value);
20:     }
21:   }, [value]);
22:
23:   const handleValueChange = (newValue) => {
24:     setSelectedValue(newValue);
25:     if (onValueChange) {
26:       onValueChange(newValue);
27:     }
28:   };
29:
30:   return (
31:     <TabsContext.Provider
32:       value={{ value: selectedValue, onValueChange: handleValueChange }}
33:     >
34:       <div className={className}>{children}</div>
35:     </TabsContext.Provider>
36:   );
37: };
38:
39: const TabsList = React.forwardRef(({ className = "", ...props }, ref) => (
40:   <div
41:     ref={ref}
42:     className={`inline-flex h-10 items-center justify-center rounded-md bg-muted p-1 text-muted-foreground`}
43:     {...props}
44:   />
45: ));
46: TabsList.displayName = "TabsList";
47:
48: const TabsTrigger = React.forwardRef(
49:   ({ className = "", value, ...props }, ref) => {
50:     const { value: selectedValue, onValueChange } =
51:       React.useContext(TabsContext);
52:     const isSelected = selectedValue === value;
53:
54:     return (
55:       <button
56:         ref={ref}
57:         className={`inline-flex items-center justify-center whitespace-nowrap rounded-sm px-3 py-1.5 text-s`}
58:         isSelected
59:         ? "bg-background text-foreground shadow-sm"
60:         : "hover:bg-background/50"
61:       > ${className} `
62:       onClick={() => onValueChange(value)}
63:       {...props}
64:     />
65:   );

```

```

66:   }
67: );
68: TabsTrigger.displayName = "TabsTrigger";
69:
70: const TabsContent = React.forwardRef(
71:   ({ className = "", value, ...props }, ref) => {
72:     const { value: selectedValue } = React.useContext(TabsContext);
73:
74:     if (selectedValue !== value) {
75:       return null;
76:     }
77:
78:     return (
79:       <div
80:         ref={ref}
81:         className={`mt-2 ring-offset-background focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring`}
82:         {...props}
83:       />
84:     );
85:   }
86: );
87: TabsContent.displayName = "TabsContent";
88:
89: export { Tabs, TabsList, TabsTrigger, TabsContent };

```

■ File: src/components/ui/textarea.jsx

```

1: // src/components/ui/textarea.jsx
2: import * as React from "react"
3:
4: const Textarea = React.forwardRef(({ className = "", ...props }, ref) => {
5:   return (
6:     <textarea
7:       className={`flex min-h-[80px] w-full rounded-md border border-input bg-background px-3 py-2 text-sm`}
8:       ref={ref}
9:       {...props}
10:     />
11:   )
12: })
13: Textarea.displayName = "Textarea"
14:
15: export { Textarea }

```

■ File: src/hooks/useBlogGenerator.js

```

1: // src/hooks/useBlogGenerator.js
2: import { useState, useCallback } from "react";
3: import { generateBlog } from "../../services/blogApi";
4: import { calculateBlogMetrics } from "../../utils/blogMetrics";
5: import { useLocalStorage } from "../../useLocalStorage";
6: import { PROMPT_CONFIG } from "../../utils/constants";
7:
8: export function useBlogGenerator() {
9:   const [prompt, setPrompt] = useLocalStorage("blog-prompt", "");
10:  const [blog, setBlog] = useState(null);
11:  const [isLoading, setIsLoading] = useState(false);
12:  const [error, setError] = useState(null);
13:
14:  const validatePrompt = (text) => {
15:    if (!text.trim()) {
16:      return "Please enter a prompt";
17:    }
18:    if (text.trim().length < PROMPT_CONFIG.MIN_LENGTH) {
19:      return `Please provide a more detailed prompt (at least ${PROMPT_CONFIG.MIN_LENGTH} characters)`;
20:    }
21:    return null;
22:  };
23:
24:  const generate = useCallback(async () => {

```

```

25:     const validationError = validatePrompt(prompt);
26:     if (validationError) {
27:         setError(validationError);
28:         return { success: false, error: validationError };
29:     }
30:
31:     setIsLoading(true);
32:     setError(null);
33:     setBlog(null);
34:
35:     const startTime = Date.now();
36:
37:     try {
38:         const data = await generateBlog(prompt);
39:         const generationTime = (Date.now() - startTime) / 1000;
40:         const metrics = calculateBlogMetrics(data.blog, generationTime);
41:
42:         const blogData = {
43:             content: data.blog,
44:             metrics,
45:             timestamp: new Date().toISOString(),
46:         };
47:
48:         setBlog(blogData);
49:         return { success: true, blog: blogData };
50:     } catch (err) {
51:         const errorMessage =
52:             err instanceof Error ? err.message : "Failed to generate blog";
53:         setError(errorMessage);
54:         return { success: false, error: errorMessage };
55:     } finally {
56:         setIsLoading(false);
57:     }
58: }, [prompt]);
59:
60: const reset = useCallback(() => {
61:     setPrompt("");
62:     setBlog(null);
63:     setError(null);
64: }, [setPrompt]);
65:
66: const copyToClipboard = useCallback(async () => {
67:     if (!blog) return { success: false };
68:
69:     try {
70:         await navigator.clipboard.writeText(blog.content);
71:         return { success: true };
72:     } catch (err) {
73:         return { success: false, error: "Failed to copy to clipboard" };
74:     }
75: }, [blog]);
76:
77: return {
78:     prompt,
79:     setPrompt,
80:     blog,
81:     isLoading,
82:     error,
83:     generate,
84:     reset,
85:     copyToClipboard,
86: };
87: }

```

■ File: src/hooks/useLocalStorage.js

```

1: // src/hooks/useLocalStorage.js
2: import { useState, useEffect } from "react";
3:
4: /**
5:  * Hook to sync state with localStorage

```



```

6:  * @param {string} key - localStorage key
7:  * @param {*} initialValue - Initial value if key doesn't exist
8:  * @returns {[any, Function]} [value, setValue]
9:  */
10: export function useLocalStorage(key, initialValue) {
11:   const [storedValue, setStoredValue] = useState(() => {
12:     try {
13:       const item = window.localStorage.getItem(key);
14:       return item ? JSON.parse(item) : initialValue;
15:     } catch (error) {
16:       console.error("Error reading from localStorage:", error);
17:       return initialValue;
18:     }
19:   });
20:
21:   const setValue = (value) => {
22:     try {
23:       const valueToStore =
24:         value instanceof Function ? value(storedValue) : value;
25:       setStoredValue(valueToStore);
26:
27:       if (
28:         valueToStore === null ||
29:         valueToStore === undefined ||
30:         valueToStore === ""
31:       ) {
32:         window.localStorage.removeItem(key);
33:       } else {
34:         window.localStorage.setItem(key, JSON.stringify(valueToStore));
35:       }
36:     } catch (error) {
37:       console.error("Error writing to localStorage:", error);
38:     }
39:   };
40:
41:   return [storedValue, setValue];
42: }

```

■ File: src/main.jsx

```

1: import { StrictMode } from 'react'
2: import { createRoot } from 'react-dom/client'
3: import './index.css'
4: import App from './App.jsx'
5:
6: createRoot(document.getElementById('root')).render(
7:   <StrictMode>
8:     <App />
9:   </StrictMode>,
10: )

```

■ File: src/services/blogApi.js

```

1: // src/services/blogApi.js
2: import { API_CONFIG } from '../utils/constants';
3:
4: /**
5:  * Generate blog using API
6:  * @param {string} prompt - User prompt
7:  * @returns {Promise<Object>} API response
8:  */
9: export async function generateBlog(prompt) {
10:   const response = await fetch(API_CONFIG.ENDPOINT, {
11:     method: 'POST',
12:     headers: {
13:       'Content-Type': 'application/json',
14:       'Accept': 'application/json',
15:     },
16:     body: JSON.stringify({ prompt: prompt.trim() }),

```

```

17: });
18:
19: const data = await response.json();
20:
21: if (!response.ok) {
22:     throw new Error(data.error || `Failed to generate blog (${response.status})`);
23: }
24:
25: if (!data.blog || data.blog.trim().length === 0) {
26:     throw new Error('Generated blog is empty. Please try again.');
```

```

27: }
28:
29: return data;
30: }
31:
32: /**
33:  * Check API health
34:  * @returns {Promise<Object>} Health status
35:  */
36: export async function checkApiHealth() {
37:     const response = await fetch(API_CONFIG.ENDPOINT, {
38:         method: 'GET',
39:     });
40:
41:     return response.json();
42: }
```

■ File: src\utils\blogMetrics.js

```

1: // src/utils/blogMetrics.js
2:
3: /**
4:  * Calculate blog content metrics
5:  * @param {string} content - The blog content
6:  * @param {number} generationTime - Time taken to generate (in seconds)
7:  * @returns {Object} Metrics object
8:  */
9: export function calculateBlogMetrics(content, generationTime) {
10:     const words = content
11:         .trim()
12:         .split(/\s+/)
13:         .filter((word) => word.length > 0);
14:     const wordCount = words.length;
15:     const characterCount = content.length;
16:     const readingTime = Math.ceil(wordCount / 200); // 200 words per minute average
17:
18:     return {
19:         wordCount,
20:         characterCount,
21:         readingTime,
22:         generationTime: generationTime.toFixed(1),
23:     };
24: }
25:
26: /**
27:  * Download blog as markdown file
28:  * @param {string} content - Blog content
29:  */
30: export function downloadBlog(content) {
31:     const element = document.createElement("a");
32:     const file = new Blob([content], { type: "text/markdown;charset=utf-8" });
33:     element.href = URL.createObjectURL(file);
34:     element.download = `blog-${new Date().toISOString().split("T")[0]}.md`;
35:     document.body.appendChild(element);
36:     element.click();
37:     document.body.removeChild(element);
38:     URL.revokeObjectURL(element.href);
39: }
```

■ File: src\utils\constants.js

```
=====
1: // src/utils/constants.js
2:
3: export const EXAMPLE_PROMPTS = [
4:   "The impact of artificial intelligence on modern healthcare",
5:   "Sustainable living practices for urban environments",
6:   "The future of remote work and digital nomadism",
7:   "Blockchain technology and its real-world applications"
8: ];
9:
10: export const FEATURES = [
11:   {
12:     title: '1000+ Words',
13:     description: 'Comprehensive blog posts with rich content',
14:     icon: 'FileText'
15:   },
16:   {
17:     title: 'AI-Powered',
18:     description: 'Advanced language model for quality output',
19:     icon: 'Sparkles'
20:   },
21:   {
22:     title: 'Lightning Fast',
23:     description: 'Generate blogs in 5-10 seconds',
24:     icon: 'Zap'
25:   },
26:   {
27:     title: 'Pro Formatting',
28:     description: 'Perfect structure with markdown support',
29:     icon: 'CheckCircle'
30:   }
31: ];
32:
33: export const PROMPT_CONFIG = {
34:   MIN_LENGTH: 10,
35:   MAX_LENGTH: 1000,
36:   PLACEHOLDER: "Example: Write a comprehensive guide about the benefits of meditation, including scientific
37: };
38:
39: export const API_CONFIG = {
40:   ENDPOINT: import.meta.env.VITE_API_ENDPOINT || '/api/generate-blog',
41:   TIMEOUT: 30000
42: };
-----
```

■ File: tailwind.config.js

```
=====
1: /** @type {import('tailwindcss').Config} */
2: export default {
3:   content: [
4:     "./index.html",
5:     "./src/**/*..{js,ts,jsx,tsx}",
6:   ],
7:   theme: {
8:     extend: {
9:       colors: {
10:         border: "hsl(214.3 31.8% 91.4%)",
11:         input: "hsl(214.3 31.8% 91.4%)",
12:         ring: "hsl(221.2 83.2% 53.3%)",
13:         background: "hsl(0 0% 100%)",
14:         foreground: "hsl(222.2 84% 4.9%)",
15:         primary: {
16:           DEFAULT: "hsl(221.2 83.2% 53.3%)",
17:           foreground: "hsl(210 40% 98%)",
18:         },
19:         secondary: {
20:           DEFAULT: "hsl(210 40% 96.1%)",
21:           foreground: "hsl(222.2 47.4% 11.2%)",
22:         },
23:         destructive: {
24:           DEFAULT: "hsl(0 84.2% 60.2%)",

```

```
25:         foreground: "hsl(210 40% 98%)",
26:     },
27:     muted: {
28:         DEFAULT: "hsl(210 40% 96.1%)",
29:         foreground: "hsl(215.4 16.3% 46.9%)",
30:     },
31:     accent: {
32:         DEFAULT: "hsl(210 40% 96.1%)",
33:         foreground: "hsl(222.2 47.4% 11.2%)",
34:     },
35:     card: {
36:         DEFAULT: "hsl(0 0% 100%)",
37:         foreground: "hsl(222.2 84% 4.9%)",
38:     },
39: },
40: borderRadius: {
41:     lg: "0.5rem",
42:     md: "calc(0.5rem - 2px)",
43:     sm: "calc(0.5rem - 4px)",
44: },
45: },
46: },
47:   plugins: [],
48: }
```

■ File: vite.config.js

```
=====
1: import { defineConfig } from 'vite'
2: import react from '@vitejs/plugin-react'
3:
4: export default defineConfig({
5:   plugins: [react()],
6:   server: {
7:     port: 3000,
8:     proxy: {
9:       "/api": {
10:        target: "http://localhost:5000",
11:        changeOrigin: true,
12:        secure: false
13:      }
14:    }
15:  }
16: })
=====
```