09-05-25 - React Project - React Color Palette Generator

JOSHNA ACSHA S

APPRENTICE - CPRIME

EMP ID: B2FFE7YKC

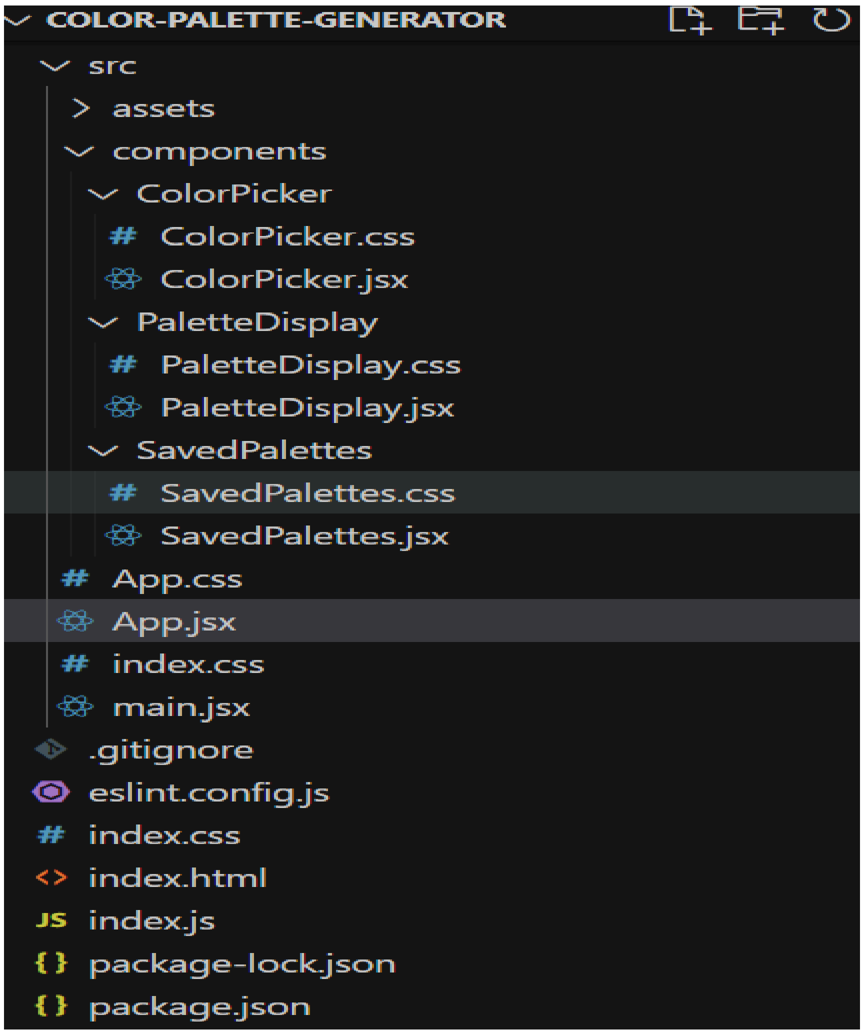
Project Overview

The **Color Palette Generator** is a responsive and intuitive React-based web application that allows users to create, preview, and manage custom color palettes. Built using **React with Vite** for faster development, this project emphasizes a clean UI and modular architecture, leveraging React functional components and hooks.

Project Structure

Here's how your project files should be organized:

```
color-palette-generator/
├── public/
│   ├── index.html
│   └── favicon.ico
├── src/
│   ├── components/
│   │   ├── ColorPicker/
│   │   │   ├── ColorPicker.js
│   │   │   └── ColorPicker.css
│   │   ├── PaletteDisplay/
│   │   │   ├── PaletteDisplay.js
│   │   │   └── PaletteDisplay.css
│   │   └── SavedPalettes/
│   │       ├── SavedPalettes.js
│   │       └── SavedPalettes.css
│   ├── App.js
│   ├── App.css
│   ├── index.js
│   └── index.css
├── package.json
└── README.md
```



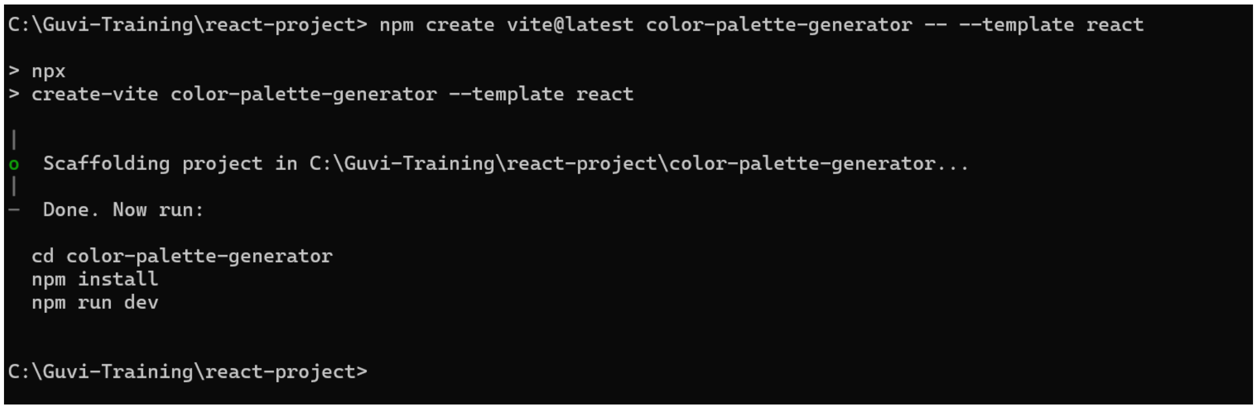
```
COLOR-PALETTE-GENERATOR
  src
    assets
    components
      ColorPicker
        ColorPicker.css
        ColorPicker.jsx
      PaletteDisplay
        PaletteDisplay.css
        PaletteDisplay.jsx
      SavedPalettes
        SavedPalettes.css
        SavedPalettes.jsx
    App.css
    App.jsx
    index.css
    main.jsx
  .gitignore
  eslint.config.js
  index.css
  index.html
  index.js
  package-lock.json
  package.json
```

Setting Up the Project

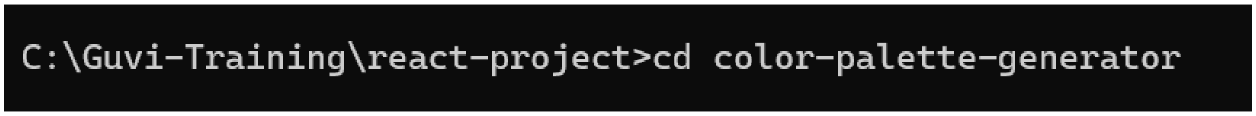
Using Vite (Faster Development)

Create a new React application with Vite:

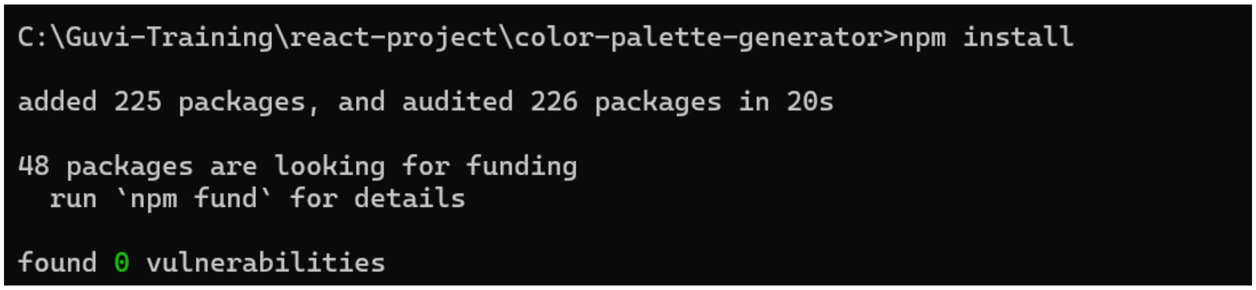
```
npm create vite@latest color-palette-generator -- --template react
```


```
C:\Guvi-Training\react-project> npm create vite@latest color-palette-generator -- --template react

> npx
> create-vite color-palette-generator --template react

|
o  Scaffolding project in C:\Guvi-Training\react-project\color-palette-generator...
|
—  Done. Now run:

  cd color-palette-generator
  npm install
  npm run dev


C:\Guvi-Training\react-project>
```

1. Navigate to the project folder:
   cd color-palette-generator

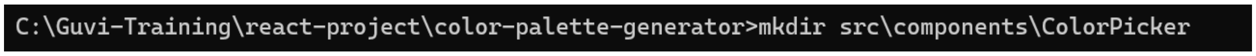

```
C:\Guvi-Training\react-project>cd color-palette-generator
```

2. Install dependencies:
   npm install

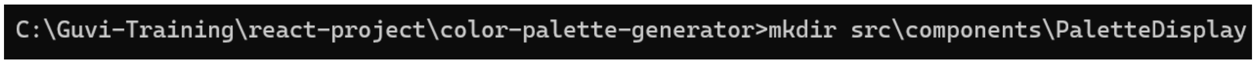

```
C:\Guvi-Training\react-project\color-palette-generator>npm install

added 225 packages, and audited 226 packages in 20s

48 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

3. Create the component folders:

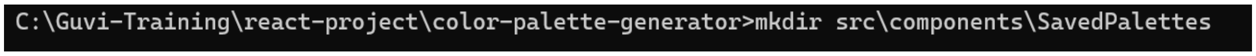
   mkdir src\components\ColorPicker

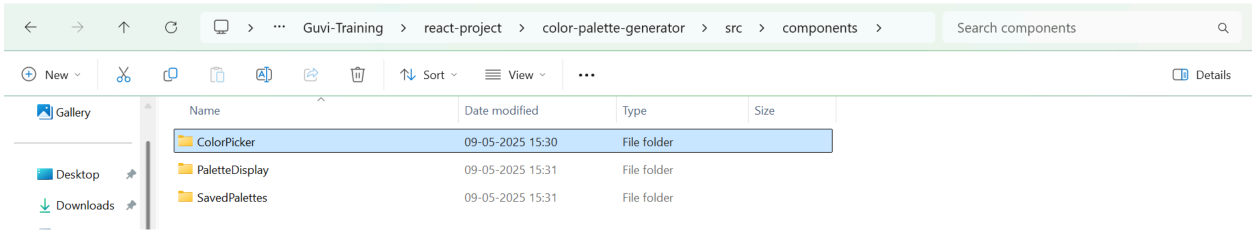

```
C:\Guvi-Training\react-project\color-palette-generator>mkdir src\components\ColorPicker
```

   mkdir src\components\PaletteDisplay


```
C:\Guvi-Training\react-project\color-palette-generator>mkdir src\components\PaletteDisplay
```

   mkdir src\components\SavedPalettes


```
C:\Guvi-Training\react-project\color-palette-generator>mkdir src\components\SavedPalettes
```

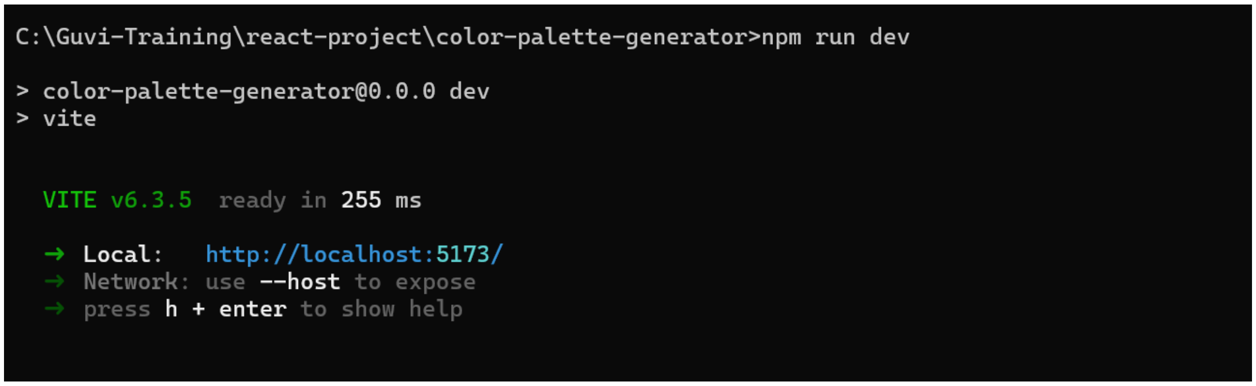

Guvi-Training > react-project > color-palette-generator > src > components >
Search components
New
Sort
View
Details
Gallery
Desktop
Downloads

Name	Date modified	Type	Size
ColorPicker	09-05-2025 15:30	File folder	
PaletteDisplay	09-05-2025 15:31	File folder	
SavedPalettes	09-05-2025 15:31	File folder	

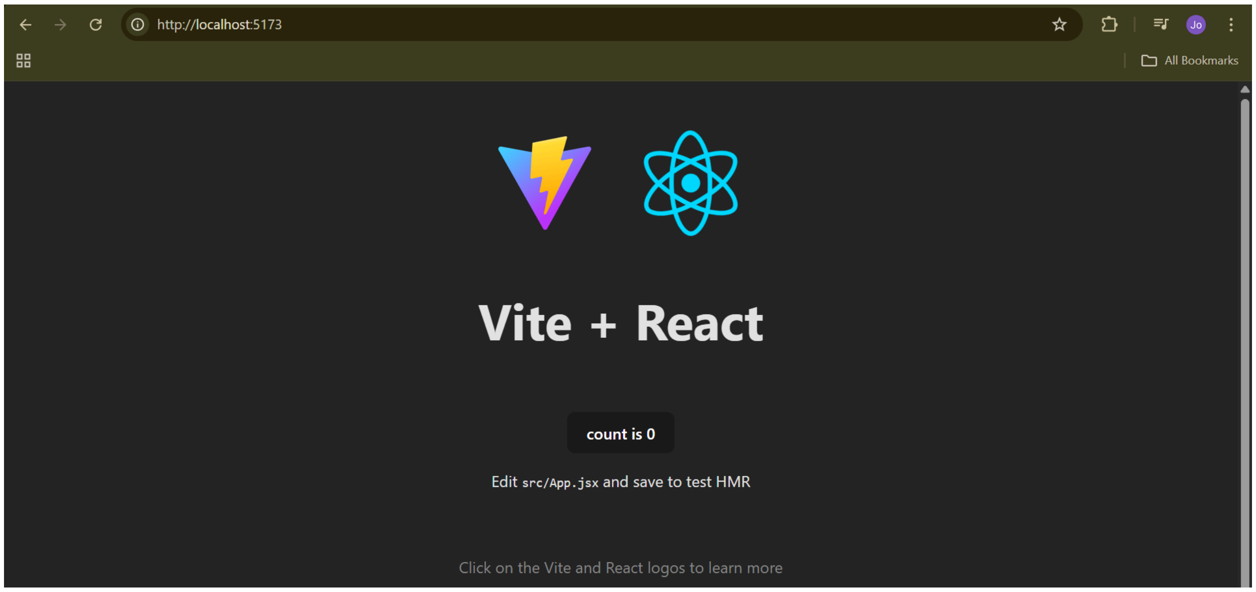
4. Copy the provided files to their respective locations.

   Start the development server:

   npm run dev



```
C:\Guvi-Training\react-project\color-palette-generator>npm run dev

> color-palette-generator@0.0.0 dev
> vite


  VITE v6.3.5  ready in 255 ms

  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h + enter to show help
```


http://localhost:5173
All Bookmarks
Vite + React
count is 0
Edit src/App.jsx and save to test HMR
Click on the Vite and React logos to learn more

Additional Notes

- The application uses React's functional components with hooks
- React 16.8+ is required for hooks support
- Make sure to import the CSS files in each component

- The project doesn't require any additional third-party dependencies
- The application is responsive and works on mobile devices

Features

1. **Color Selection**
   - Color picker input
   - HEX code input
   - Preset colors for quick selection
2. **Palette Management**
   - Add up to 5 colors per palette
   - Remove individual colors
   - Preview the palette
   - Save palettes with custom names
3. **Saved Palettes**
   - View all saved palettes
   - Load palettes back to the editor
   - Delete unwanted palettes

Code:

App.js

```
import React, { useState } from 'react';
import './App.css';
import ColorPicker from './components/ColorPicker/ColorPicker';
import PaletteDisplay from './components/PaletteDisplay/PaletteDisplay';
import SavedPalettes from './components/SavedPalettes/SavedPalettes';

function App() {
  const [colors, setColors] = useState([]);
  const [savedPalettes, setSavedPalettes] = useState([]);

  const handleColorSelected = (color) => {
    if (colors.length < 5) {
      setColors([...colors, color]);
    }
  };

  const handleSavePalette = (name) => {
    if (colors.length > 0) {
      const newPalette = {
```

```
      id: Date.now(), // Unique ID for React keys
      name,
      colors: [...colors]
    };

    setSavedPalettes([...savedPalettes, newPalette]);
    setColors([]); // Clear current palette
  }
};

const handleSelectPalette = (palette) => {
  setColors([...palette.colors]);
};

const handleDeletePalette = (paletteId) => {
  setSavedPalettes(savedPalettes.filter(palette => palette.id !== paletteId));
};

return (
  <div className="app-container">
    <h1>Color Palette Generator</h1>
    <ColorPicker onColorSelected={handleColorSelected} />
    <PaletteDisplay
      colors={colors}
      onSave={handleSavePalette}
      onRemoveColor={(index) => {
        const newColors = [...colors];
        newColors.splice(index, 1);
        setColors(newColors);
      }}
    />
    <SavedPalettes
      palettes={savedPalettes}
      onSelect={handleSelectPalette}
      onDelete={handleDeletePalette}
    />
  </div>
);
}

export default App;
```

App.css

```
/* App.css */
```

```css
.app-container {
  max-width: 800px;
  margin: 0 auto;
  padding: 20px;
  font-family: Arial, sans-serif;
}

h1 {
  color: #333;
  text-align: center;
  margin-bottom: 30px;
}

/* Set global styles */
* {
  box-sizing: border-box;
}

body {
  margin: 0;
  padding: 0;
  background-color: #f5f5f5;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
}
```

ColorPicker.jsx

```jsx
import React, { useState } from 'react';
import './ColorPicker.css';

const ColorPicker = ({ onColorSelected }) => {
  const [selectedColor, setSelectedColor] = useState('#3366FF');

  const presetColors = [
    '#FF6633', '#FFB399', '#FF33FF', '#FFFF99', '#00B3E6',
    '#E6B333', '#3366E6', '#999966', '#99FF99', '#B34D4D',
    '#80B300', '#809900', '#E6B3B3', '#6680B3', '#66991A',
    '#FF99E6', '#CCFF1A', '#FF1A66', '#E6331A', '#33FFCC'
  ];

  const isValidHexColor = (color) => {
    const regex = /^#([A-Fa-f0-9]{6})$/;
    return regex.test(color);
```

```
};

const handleColorChange = (e) => {
  setSelectedColor(e.target.value);
};

const handleInputChange = (e) => {
  const value = e.target.value;
  setSelectedColor(value);
};

const emitColor = () => {
  if (isValidHexColor(selectedColor)) {
    onColorSelected(selectedColor);
  }
};

const selectPreset = (color) => {
  setSelectedColor(color);
  onColorSelected(color);
};

return (
  <div className="color-picker-container">
    <div className="color-input-group">
      <input
        type="color"
        value={selectedColor}
        onChange={handleColorChange}
        className="color-input"
      />
      <input
        type="text"
        value={selectedColor}
        onChange={handleInputChange}
        placeholder="#RRGGBB"
        pattern="^#([A-Fa-f0-9]{6})$"
        className="color-text-input"
      />
      <button onClick={emitColor} className="add-button">Add Color</button>
    </div>

    <div className="presets">
      <h3>Quick Colors</h3>
```

```
        <div className="preset-colors">
          {presetColors.map((color, index) => (
            <button
              key={index}
              className="preset-color"
              style={{ backgroundColor: color }}
              onClick={() => selectPreset(color)}
              aria-label={`Select color ${color}`}
            />
          ))}
        </div>
      </div>
    </div>
  );
};

export default ColorPicker;
```

ColorPicker.css

```
/* ColorPicker.css */
.color-picker-container {
  background-color: #f5f5f5;
  padding: 20px;
  border-radius: 8px;
  margin-bottom: 20px;
  box-shadow: 0 1px 3px rgba(0, 0, 0, 0.1);
}

.color-input-group {
  display: flex;
  gap: 10px;
  margin-bottom: 15px;
}

.color-input {
  width: 60px;
  height: 40px;
  padding: 0;
  border: 1px solid #ccc;
  cursor: pointer;
}

.color-text-input {
  flex: 1;
  padding: 8px;
  border: 1px solid #ccc;
  border-radius: 4px;
  font-family: monospace;
}

.add-button {
  padding: 8px 15px;
  background-color: #4CAF50;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.2s;
}

.add-button:hover {
  background-color: #45a049;
}

.presets {
  margin-top: 15px;
}

.presets h3 {
  margin-top: 0;
  margin-bottom: 10px;
  font-size: 16px;
  color: #333;
}

.preset-colors {
  display: flex;
  flex-wrap: wrap;
  gap: 10px;
}

.preset-color {
  width: 30px;
  height: 30px;
  border-radius: 50%;
  border: 1px solid #ddd;
  cursor: pointer;
  transition: transform 0.2s;
```

```
}

.preset-color:hover {
  transform: scale(1.1);
}
```

PaletteDisplay.jsx

```
import React, { useState } from 'react';
import './PaletteDisplay.css';

const PaletteDisplay = ({ colors, onSave, onRemoveColor }) => {
  const [paletteName, setPaletteName] = useState('');

  const handleSavePalette = () => {
    if (paletteName && colors.length > 0) {
      onSave(paletteName);
      setPaletteName('');
    }
  };

  if (colors.length === 0) {
    return (
      <div className="no-colors">
        <p>Add colors to start building your palette</p>
      </div>
    );
  }

  return (
    <div className="palette-display">
      <h2>Current Palette</h2>
      <div className="colors-container">
        {colors.map((color, index) => (
          <div
            key={index}
            className="color-box"
            style={{ backgroundColor: color }}
          >
            <div className="color-details">
              <span className="color-value">{color}</span>
              <button
                className="remove-button"
                onClick={() => onRemoveColor(index)}
                aria-label={`Remove color ${color}`}
              >
                ×
              </button>
            </div>
          </div>
        ))}
      </div>

      <div className="palette-preview">
        <h3>Preview</h3>
        <div className="preview-sample">
          {colors.map((color, index) => (
            <div
              key={index}
              className="preview-color"
              style={{ backgroundColor: color }}
            />
          ))}
        </div>
      </div>

      <div className="save-palette">
        <input
          type="text"
          value={paletteName}
          onChange={(e) => setPaletteName(e.target.value)}
          placeholder="Enter palette name"
          className="palette-name-input"
        />
        <button
          onClick={handleSavePalette}
          disabled={!paletteName}
          className="save-button"
        >
          Save Palette
        </button>
      </div>
    </div>
  );
};

export default PaletteDisplay;
```

PaletteDisplay.css

```css
/* PaletteDisplay.css */
.palette-display {
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
  margin-bottom: 20px;
}

.palette-display h2 {
  margin-top: 0;
  color: #333;
  font-size: 20px;
}

.colors-container {
  display: flex;
  flex-wrap: wrap;
  gap: 10px;
  margin-bottom: 20px;
}

.color-box {
  width: calc(20% - 8px);
  aspect-ratio: 1;
  border-radius: 8px;
  position: relative;
  overflow: hidden;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

@media (max-width: 600px) {
  .color-box {
    width: calc(33.333% - 8px);
  }
}

.color-details {
  position: absolute;
  bottom: 0;
  left: 0;
  right: 0;
  background-color: rgba(0,0,0,0.7);
  color: white;
  padding: 5px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  font-size: 12px;
}

.remove-button {
  background: none;
  border: none;
  color: white;
  font-size: 16px;
  cursor: pointer;
  padding: 0;
  width: 20px;
  height: 20px;
  display: flex;
  align-items: center;
  justify-content: center;
}

.remove-button:hover {
  background-color: rgba(255,255,255,0.2);
  border-radius: 50%;
}

.palette-preview {
  margin-bottom: 20px;
}

.palette-preview h3 {
  margin-top: 0;
  margin-bottom: 10px;
  font-size: 16px;
  color: #333;
}

.preview-sample {
  height: 40px;
  display: flex;
  border-radius: 4px;
  overflow: hidden;
  box-shadow: 0 1px 3px rgba(0,0,0,0.1);
}
```

```
.preview-color {
  flex: 1;
}

.save-palette {
  display: flex;
  gap: 10px;
}

.palette-name-input {
  flex: 1;
  padding: 8px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

.save-button {
  padding: 8px 15px;
  background-color: #3498db;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.2s;
}

.save-button:hover:not([disabled]) {
  background-color: #2980b9;
}

.save-button[disabled] {
  background-color: #95a5a6;
  cursor: not-allowed;
}

.no-colors {
  background-color: #f8f8f8;
  padding: 30px;
  text-align: center;
  border-radius: 8px;
  color: #666;
  box-shadow: 0 1px 3px rgba(0,0,0,0.1);
  margin-bottom: 20px;
}
```

SavedPalattes.jsx

```
import React from 'react';
import './SavedPalettes.css';

const SavedPalettes = ({ palettes, onSelect, onDelete }) => {
  if (palettes.length === 0) {
    return (
      <div className="saved-palettes">
        <h2>Saved Palettes</h2>
        <div className="no-palettes">
          <p>No saved palettes yet</p>
        </div>
      </div>
    );
  }

  return (
    <div className="saved-palettes">
      <h2>Saved Palettes</h2>
      <div className="palettes-list">
        {palettes.map(palette => (
          <div
            key={palette.id}
            className="palette-item"
            onClick={() => onSelect(palette)}
          >
            <h3>{palette.name}</h3>
            <div className="palette-preview">
              {palette.colors.map((color, index) => (
                <div
                  key={index}
                  className="preview-color"
                  style={{ backgroundColor: color }}
                />
              ))}
            </div>
            <div className="palette-colors">
              {palette.colors.map((color, index) => (
                <span key={index} className="color-badge">
                  {color}
                </span>
              ))}
            </div>
            <div className="palette-actions">
              <button
                className="load-button"
                onClick={(e) => {
                  e.stopPropagation();
                  onSelect(palette);
                }}
              >
                Load Palette
              </button>
              <button
                className="delete-button"
                onClick={(e) => {
                  e.stopPropagation();
                  onDelete(palette.id);
                }}
              >
                Delete
              </button>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
};

export default SavedPalettes;
```

SavedPalettes.css

```
/* SavedPalettes.css */
.saved-palettes {
  background-color: #fff;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

.saved-palettes h2 {
  margin-top: 0;
  color: #333;
  margin-bottom: 20px;
  font-size: 20px;
```

```
}

.no-palettes {
  text-align: center;
  padding: 20px;
  background-color: #f8f8f8;
  border-radius: 6px;
  color: #666;
}

.palettes-list {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(250px, 1fr));
  gap: 15px;
}

@media (max-width: 600px) {
  .palettes-list {
    grid-template-columns: 1fr;
  }
}

.palette-item {
  background-color: #f8f8f8;
  border-radius: 8px;
  padding: 15px;
  cursor: pointer;
  transition: transform 0.2s;
  box-shadow: 0 1px 3px rgba(0,0,0,0.1);
}

.palette-item:hover {
  transform: translateY(-3px);
  box-shadow: 0 4px 8px rgba(0,0,0,0.1);
}

.palette-item h3 {
  margin-top: 0;
  margin-bottom: 10px;
  color: #333;
  font-size: 16px;
}

.palette-preview {
  height: 30px;
  display: flex;
  border-radius: 4px;
  overflow: hidden;
  margin-bottom: 10px;
  box-shadow: 0 1px 2px rgba(0,0,0,0.1);
}

.preview-color {
  flex: 1;
}

.palette-colors {
  display: flex;
  flex-wrap: wrap;
  gap: 5px;
  margin-bottom: 15px;
  font-size: 12px;
}

.color-badge {
  background-color: rgba(0,0,0,0.05);
  padding: 2px 6px;
  border-radius: 4px;
  font-family: monospace;
}

.palette-actions {
  display: flex;
  gap: 10px;
}

.load-button, .delete-button {
  flex: 1;
  padding: 8px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-size: 12px;
  transition: background-color 0.2s;
}

.load-button {
  background-color: #3498db;
  color: white;
}

.load-button:hover {
  background-color: #2980b9;
}

.delete-button {
  background-color: #e74c3c;
  color: white;
}

.delete-button:hover {
  background-color: #c0392b;
}
```

index.js

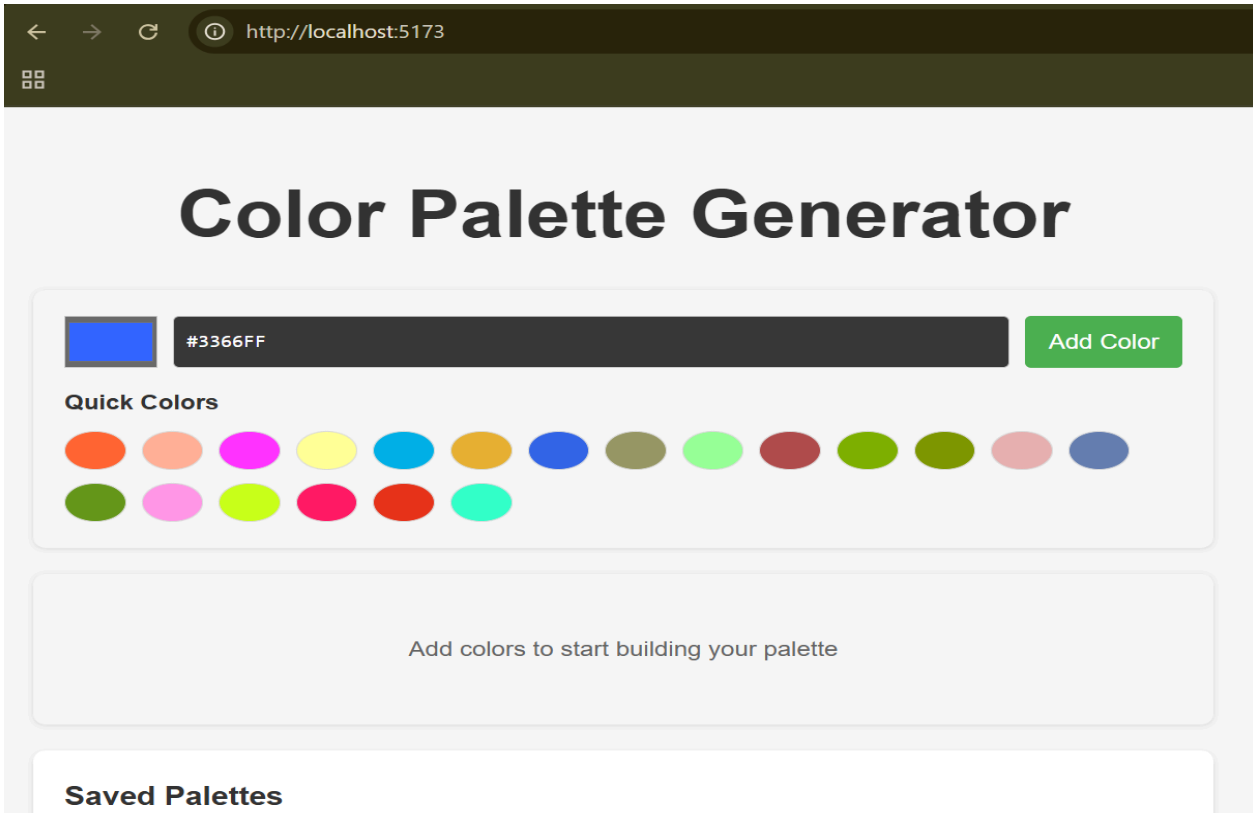
```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

index.css

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  background-color: #f0f2f5;
  padding: 20px;
}

code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

Output:

Landing Page



http://localhost:5173

Color Palette Generator

#3366FF

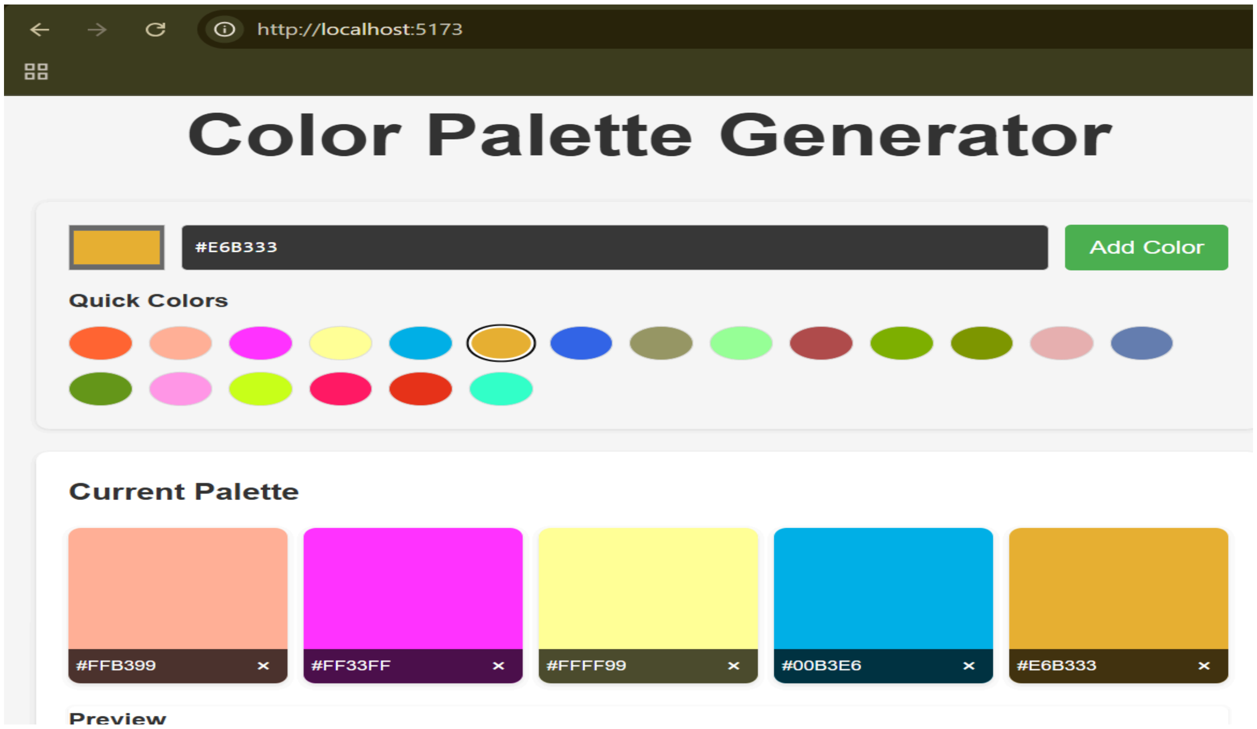
Add Color

Quick Colors

Add colors to start building your palette

Saved Palettes

Create Color Palette



Color Palette Generator

#E6B333

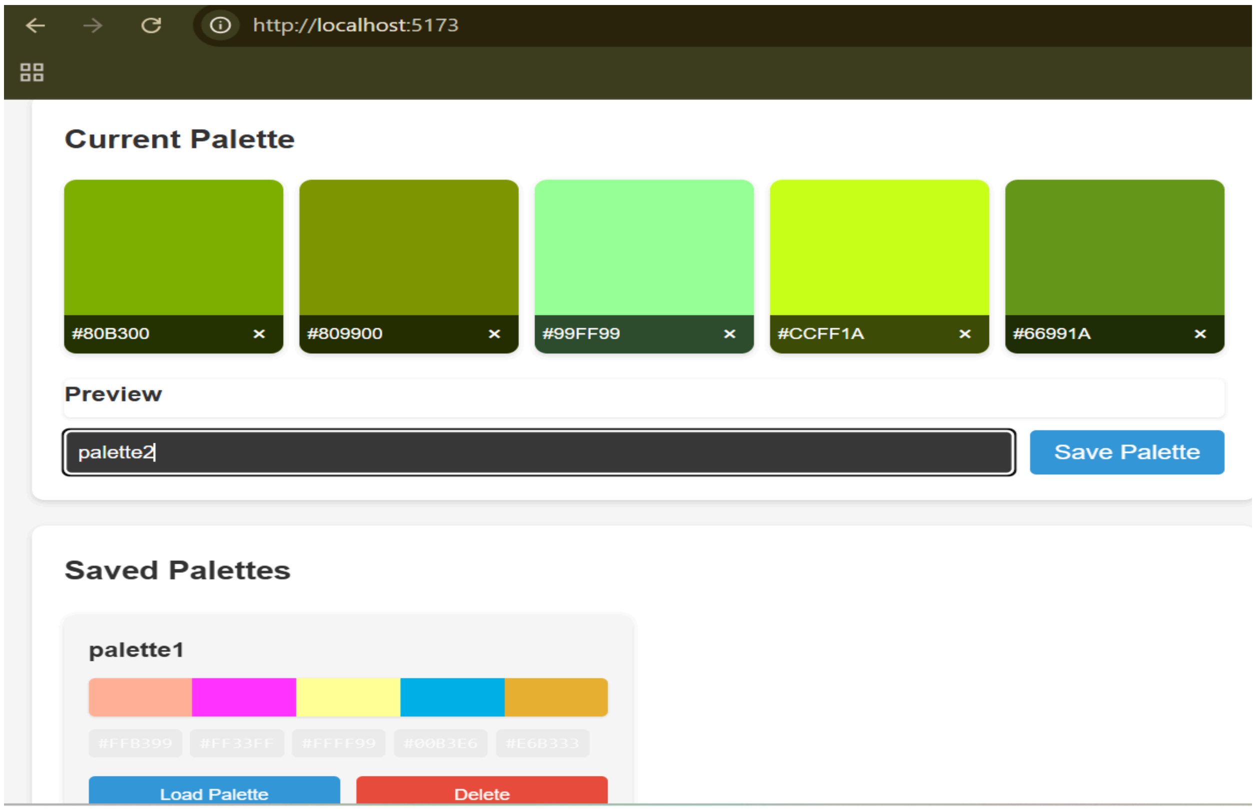
Quick Colors

Current Palette

#FFB399 #FF33FF #FFFF99 #00B3E6 #E6B333

Preview

Save Palette



Current Palette

#80B300 #809900 #99FF99 #CCFF1A #66991A

Preview

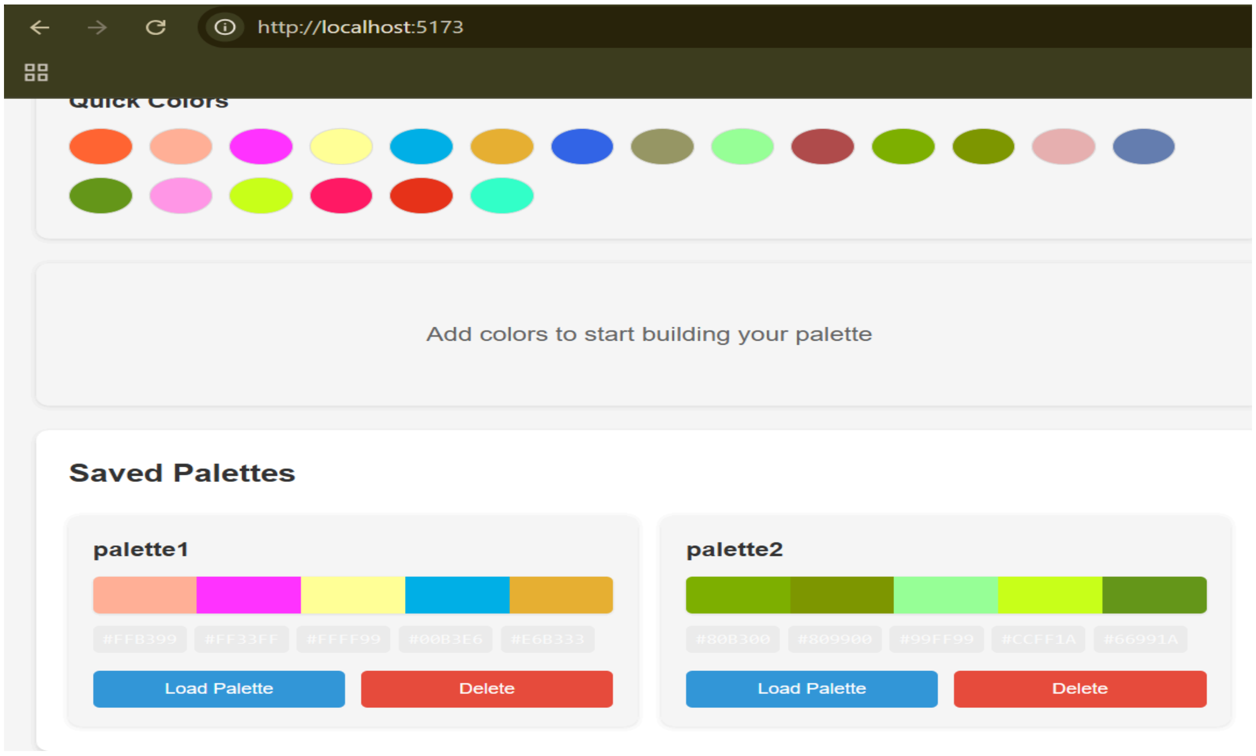
palette2

Saved Palettes

palette1

#FFB399 #FF33FF #FFFF99 #00B3E6 #E6B333

View Saved Palettes


http://localhost:5173

Quick Colors

Add colors to start building your palette

Saved Palettes

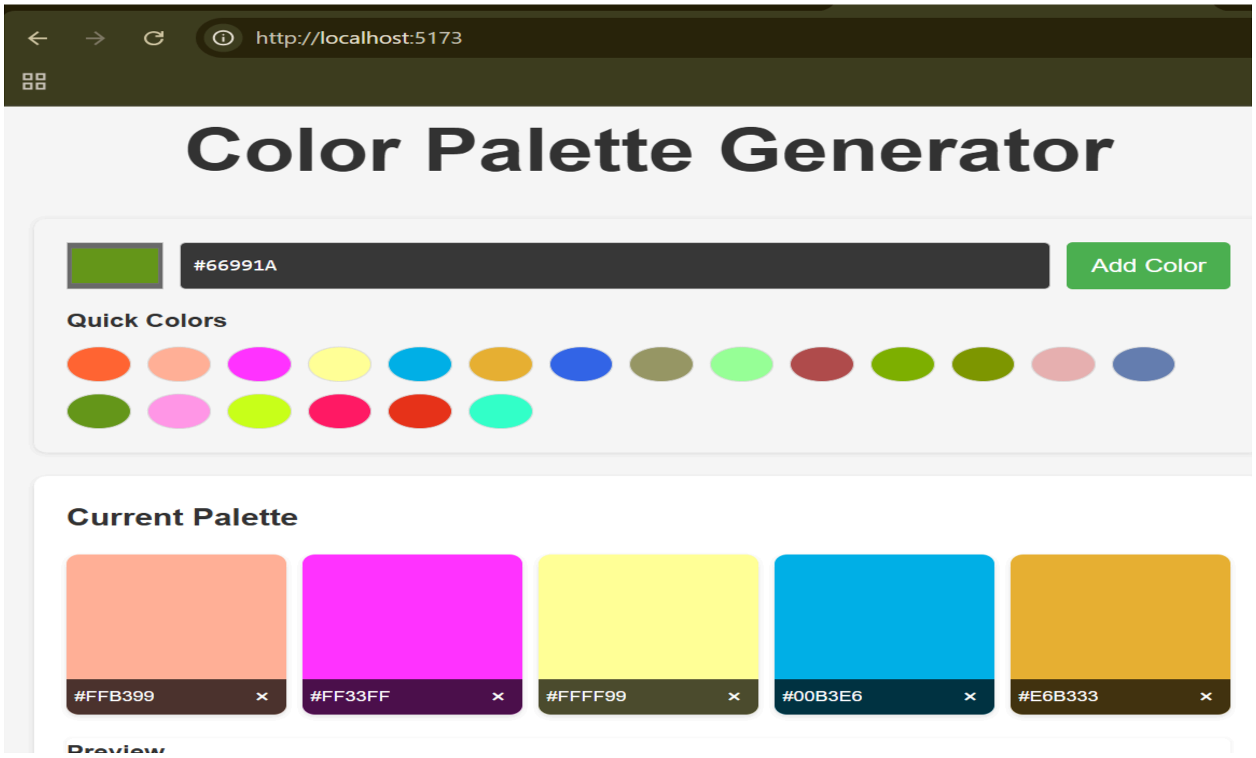
palette1

#FFB399 #FF33FF #FFFF99 #00B3E6 #E6B333

palette2

#80B300 #809900 #99FF99 #CCFF1A #66991A

Load Palette


http://localhost:5173

Color Palette Generator

#66991A

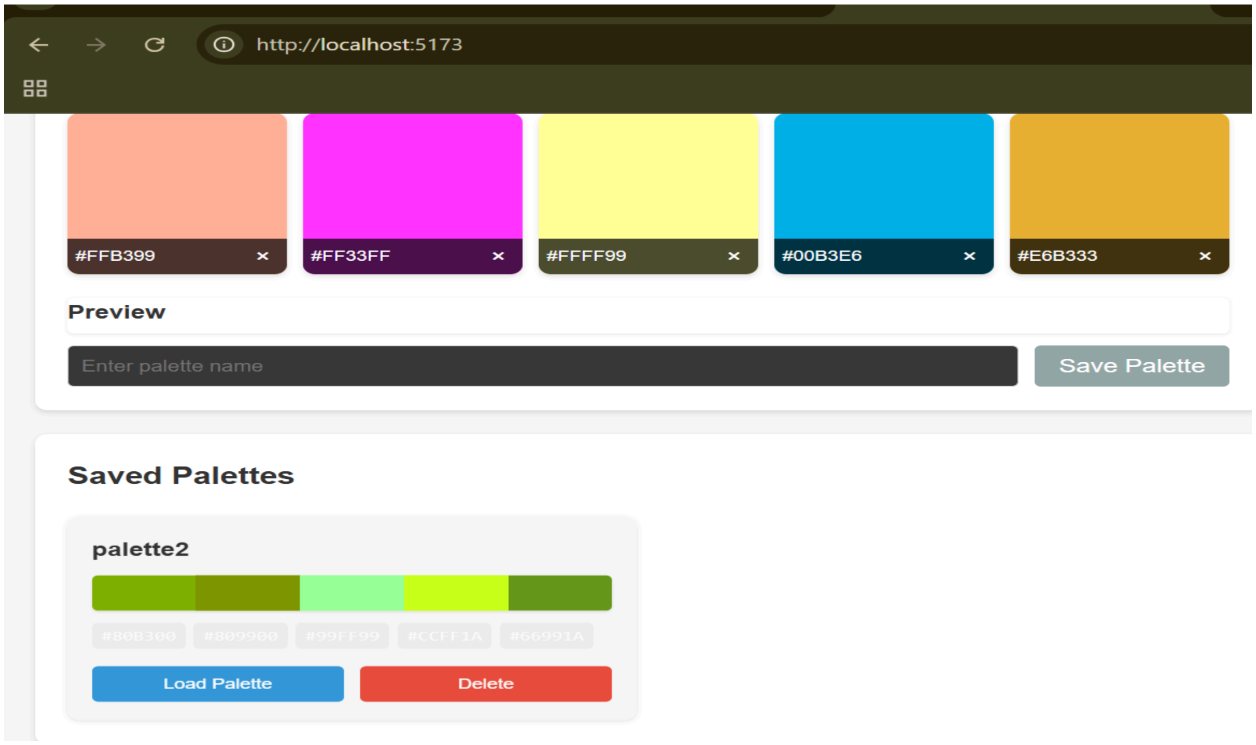
Quick Colors

Current Palette

#FFB399 ×
#FF33FF ×
#FFFF99 ×
#00B3E6 ×
#E6B333 ×

Preview

Delete Palette


http://localhost:5173
#FFB399 ×
#FF33FF ×
#FFFF99 ×
#00B3E6 ×
#E6B333 ×
Preview
Enter palette name
Save Palette
Saved Palettes
palette2
#80B300
#809900
#99FF99
#CCFF1A
#66991A
Load Palette
Delete