BlekkPro PMU Brushes and Subscription Features Guide

Overview

This guide provides comprehensive instructions for integrating specialized PMU brushes and subscription-based feature locking into your BlekkPro application. These enhancements will allow you to offer premium features exclusively to Pro subscribers while providing a compelling upgrade path for free users.

Table of Contents

Feature Overview

The BlekkPro PMU Brushes and Subscription Features include:

Specialized PMU Brushes

- **Shader Needles**: 3RS, 5RS, 7RS with realistic dispersion patterns
- **Liner Needles**: 1RL, 3RL, 5RL with precise line control
- **Magnum Needles**: 5M, 7M with flat, angled patterns
- **Specialized Tools**: Ombre lip outline, lip blender, microblade (sharp/natural), powder brow

Subscription-Based Feature Locking

- **Free Tier**: Basic shader and liner brushes
- **Pro Tier**: All specialized PMU brushes, high-resolution exports
- **Upgrade Prompts**: User-friendly prompts when accessing locked features

Integration with Layer System

- Seamless integration with the existing layer-based editing system
- Realistic brush strokes on layers with proper blending
- Before/after comparison with subscription-aware features

Module Descriptions

1. PMU Brushes Module (`pmu_brushes.js`)

This module provides realistic digital simulations of PMU needles and brushes:

- Comprehensive brush presets for all common PMU needle configurations
- Realistic stroke rendering with pressure sensitivity
- Specialized effects for ombre lips, microblading, and powder brows
- Subscription-aware brush access controls

2. Subscription Integration Module (`subscription_integration.js`)

This module integrates the PMU brushes with the layer editing system and manages subscription features:

- Unified API for accessing all features
- Subscription level management
- Upgrade prompt system for locked features
- Integration with the layer editing system

3. Testing Module (`pmu_brushes_tester.js`)

This module provides comprehensive tests for validating:

- PMU brush realism and accuracy
- Subscription-based feature locking
- Integration with the layer editing system
- Performance benchmarks

Integration with React

Basic Integration Example

```
 import React, { useEffect, useRef, useState } from 'react';
import BlekkProSubscriptionIntegration from './subscription_integration';

const BlekkProApp = () => {
  const canvasRef = useRef(null);
  const [subscriptionIntegration, setSubscriptionIntegration] = useState(null);
  const [isInitialized, setIsInitialized] = useState(false);
  const [currentBrush, setCurrentBrush] = useState(null);
  const [availableBrushes, setAvailableBrushes] = useState(null);
  const [subscription, setSubscription] = useState('free');
  const [showUpgradePrompt, setShowUpgradePrompt] = useState(false);
  const [upgradePrompt, setUpgradePrompt] = useState(null);

  // Initialize subscription integration
  useEffect(() => {
    const initialize = async () => {
      const integration = new BlekkProSubscriptionIntegration();

      const initialized = integration.initialize({
        canvasWidth: 800,
        canvasHeight: 600,
        subscription: subscription
      });

      if (initialized) {
        // Register upgrade callback
        integration.onUpgradeNeeded(prompt => {
          setUpgradePrompt(prompt);
          setShowUpgradePrompt(true);
        });

        setSubscriptionIntegration(integration);
        setIsInitialized(true);

        // Get available brushes
        const brushes = integration.getAvailableBrushes();
        setAvailableBrushes(brushes);

        // Set default brush
        const result = integration.selectBrush('basic', 'standardShader');
        if (result.success) {
          setCurrentBrush(result.brush);
        }
      }
    };
```

```
  initialize();
}, [subscription]);

// Handle brush selection
const handleBrushSelect = (category, brushId) => {
  if (!isInitialized || !subscriptionIntegration) return;

  const result = subscriptionIntegration.selectBrush(category, brushId);

  if (result.success) {
    setCurrentBrush(result.brush);
  }
};

// Handle image upload
const handleImageUpload = (event) => {
  if (!isInitialized || !subscriptionIntegration) return;

  const file = event.target.files[0];
  const image = new Image();
  image.src = URL.createObjectURL(file);

  image.onload = () => {
    // Initialize layer system with the image
    subscriptionIntegration.initLayersWithBaseImage(image);

    // Render to canvas
    subscriptionIntegration.renderToCanvas(canvasRef.current);
  };
};

// Handle brush stroke
const handleBrushStroke = (points, color) => {
  if (!isInitialized || !subscriptionIntegration) return;

  // Apply brush stroke
  subscriptionIntegration.applyBrushStroke(points, color);

  // Render to canvas
  subscriptionIntegration.renderToCanvas(canvasRef.current);
};

// Handle specialized PMU effect
const handlePMUEffect = (effectType, options) => {
  if (!isInitialized || !subscriptionIntegration) return;

  // Apply PMU effect
  const result = subscriptionIntegration.applyPMUEffect(effectType, options);

  if (result.success) {
    // Render to canvas
    subscriptionIntegration.renderToCanvas(canvasRef.current);
```

```
    }

    return result;
  };

  // Handle subscription upgrade
  const handleUpgrade = () => {
    setSubscription('pro');
    setShowUpgradePrompt(false);
  };

  // Render upgrade prompt
  const renderUpgradePrompt = () => {
    if (!showUpgradePrompt || !upgradePrompt) return null;

    return (
      <div className="upgrade-prompt">
        <h3>{upgradePrompt.title}</h3>
        <p>{upgradePrompt.message}</p>
        <ul>
          {upgradePrompt.benefits.map((benefit, index) => (
            <li key={index}>{benefit}</li>
          ))}
        </ul>
        <div className="upgrade-actions">
          <button onClick={handleUpgrade}>Upgrade Now</button>
          <button onClick={() => setShowUpgradePrompt(false)}>Maybe Later</
button>
        </div>
      </div>
    );
  };

  // Render brush selector
  const renderBrushSelector = () => {
    if (!availableBrushes) return null;

    return (
      <div className="brush-selector">
        <h3>Basic Brushes</h3>
        <div className="brush-list">
          {Object.entries(availableBrushes.basic).map(([id, brush]) => (
            <div
              key={id}
              className={`brush-item ${currentBrush && currentBrush.name ===
brush.name ? 'active' : ''}`}
              onClick={() => handleBrushSelect('basic', id)}
            >
              {brush.name}
            </div>
          ))}
        </div>
```

```
      {availableBrushes.advanced && (
        <>
          <h3>Advanced Brushes</h3>
          <div className="brush-list">
            {Object.entries(availableBrushes.advanced).map(([id, brush]) => (
              <div
                key={id}
                className={`brush-item ${currentBrush && currentBrush.name ===
brush.name ? 'active' : ''}`}
                onClick={() => handleBrushSelect('advanced', id)}
              >
                {brush.name}
              </div>
            ))}
          </div>
        </>
      )}

      {availableBrushes.locked && (
        <>
          <h3>Pro Brushes (Locked)</h3>
          <div className="brush-list locked">
            {Object.entries(availableBrushes.locked).map(([id, brush]) => (
              <div
                key={id}
                className="brush-item locked"
                onClick={() => handleBrushSelect('advanced', id)}
              >
                {brush.name}
              </div>
            ))}
          </div>
        </>
      )}
    </div>
  );
};

  return (
    <div className="blekkpro-app">
      <h1>BlekkPro PMU Tools</h1>

      <div className="subscription-status">
        Current Plan: {subscription === 'pro' ? 'Pro' : 'Free'}
        {subscription !== 'pro' && (
          <button onClick={handleUpgrade}>Upgrade to Pro</button>
        )}
      </div>

      <div className="upload-section">
        <input type="file" accept="image/*" onChange={handleImageUpload} />
```

```
      </div>

      <div className="editor-container">
        {renderBrushSelector()}

        <div className="canvas-container">
          <canvas ref={canvasRef} width="800" height="600"></canvas>
        </div>
      </div>

      {renderUpgradePrompt()}
    </div>
  );
};

export default BlekkProApp;
```

Subscription Management

Setting Up Subscription Levels

The subscription system is designed to be flexible and can be integrated with your existing payment provider:

```
// Initialize with default subscription level
const integration = new BlekkProSubscriptionIntegration();
integration.initialize({
  subscription: 'free' // or 'pro'
});

// Update subscription level when user upgrades
integration.setSubscription('pro');

// Check current subscription level
const currentSubscription = integration.getSubscription();
```

Handling Upgrade Prompts

The system provides elegant upgrade prompts when users attempt to access locked features:

```
// Register upgrade callback
integration.onUpgradeNeeded(prompt => {
  // Show upgrade prompt to user
  showUpgradePrompt(prompt);
});
```

```
// Example prompt structure
{
  title: 'Unlock Ombre Lip Blush Tools',
  message: 'Upgrade to BlekkPro to access professional ombre lip blush tools for perfect lip treatments.',
  benefits: [
    'Specialized lip outline brushes',
    'Gradient and feathering controls',
    'Realistic lip blush simulation'
  ]
}
```

PMU Brush Implementation

Available Brush Types

The PMU brushes module includes a comprehensive set of brushes:

Shader Needles

- 3RS: Round shader with 3 needles
- 5RS: Round shader with 5 needles
- 7RS: Round shader with 7 needles

Liner Needles

- 1RL: Single needle liner
- 3RL: 3-needle liner
- 5RL: 5-needle liner

Magnum Needles

- 5M: 5-needle magnum
- 7M: 7-needle magnum

Specialized Brushes

- Ombre Lip Outline: For creating perfect lip contours with feathering
- Lip Blender: For blending and filling lip areas
- Microblade Sharp: For crisp, defined brow strokes
- Microblade Natural: For softer, more natural brow strokes
- Powder Brow: For creating powder/ombre brow effects

Using PMU Brushes

```
// Get available brushes
const brushes = integration.getAvailableBrushes();

// Select a brush
const result = integration.selectBrush('advanced', 'shader3RS');

if (result.success) {
  // Brush selected successfully
  const brush = result.brush;

  // Apply brush stroke
  const points = generateStrokePoints(); // Array of {x, y, pressure} points
  integration.applyBrushStroke(points, [121, 85, 61]); // RGB color

  // Render to canvas
  integration.renderToCanvas(canvasRef.current);
}
```

Specialized PMU Effects

```
// Apply ombre lip effect
const lipResult = integration.applyPMUEffect('ombre_lip', {
  points: lipPoints, // Array of {x, y, pressure} points
  color: [224, 17, 95] // RGB color
});

// Apply microblade effect
const browResult = integration.applyPMUEffect('microblade', {
  points: browPoints, // Array of {x, y, pressure} points
  color: [61, 43, 31] // RGB color
});
```

Customization Guide

Customizing Brush Presets

You can add or modify brush presets in the `pmu_brushes.js` file:

```
// Add a new brush preset
this.brushPresets.advanced.customBrush = {
  name: "Custom Brush",
  type: "specialized",
  size: 4,
```

```
  opacity: 0.8,
  spacing: 0.2,
  scatter: 0.1,
  pressureSensitivity: 0.7,
  subscription: "pro" // or "free"
};
```

Customizing Subscription Tiers

You can modify the subscription tiers and feature availability:

```
// In pmu_brushes.js
isFeatureAvailable(featureId) {
  // Define features and their required subscription levels
  const features = {
    'basic_brushes': 'free',
    'shader_brushes': 'free',
    'liner_brushes': 'free',
    'specialized_brushes': 'pro',
    'ombre_lip': 'pro',
    'microblade': 'pro',
    'powder_brow': 'pro',
    'needle_configuration': 'pro',
    'export_high_res': 'pro',
    // Add your custom features here
    'custom_feature': 'pro'
  };

  // Rest of the method remains the same
}
```

Customizing Upgrade Prompts

You can customize the upgrade prompts for different features:

```
// In pmu_brushes.js
getUpgradePrompt(featureId) {
  // Define feature-specific upgrade prompts
  const prompts = {
    // Add your custom prompt
    'custom_feature': {
      title: 'Unlock Custom Feature',
      message: 'Upgrade to BlekkPro to access this custom feature.',
      benefits: [
        'Benefit 1',
        'Benefit 2',
        'Benefit 3'
      ]
```

```
  },
  // Rest of the prompts remain the same
 };

 // Return feature-specific prompt or default
 return prompts[featureId] || prompts.default;
}
```

Testing and Validation

The included testing module (`pmu_brushes_tester.js`) provides comprehensive tests for validating:

Running Tests

```
import BlekkProPMUBrushesTester from './pmu_brushes_tester';

// Initialize tester
const tester = new BlekkProPMUBrushesTester();
tester.initialize();

// Run all tests
const results = await tester.runAllTests();

// Get test summary
console.log(results);
```

Test Categories

1. **Brush Realism**: Tests the realism and accuracy of PMU brush strokes
2. **Subscription Controls**: Tests the subscription-based feature locking
3. **Integration**: Tests the integration with the layer editing system
4. **Performance**: Tests the performance of brush strokes and rendering

Example Test Results

```
{
 totalTests: 25,
 passedTests: 23,
 failedTests: 2,
 passRate: "92.00%",
 categories: {
  brushRealism: {
   passed: 8,
   failed: 0,
```

```
      total: 8,
      passRate: "100.00%"
    },
    subscriptionControls: {
      passed: 6,
      failed: 0,
      total: 6,
      passRate: "100.00%"
    },
    integration: {
      passed: 5,
      failed: 1,
      total: 6,
      passRate: "83.33%"
    },
    performance: {
      passed: 4,
      failed: 1,
      total: 5,
      passRate: "80.00%"
    }
  }
}
```


Conclusion

This implementation provides a comprehensive solution for adding specialized PMU brushes and subscription-based feature locking to your BlekkPro application. The modular design allows for easy customization and extension, while the testing framework ensures reliability and performance.

By offering premium features exclusively to Pro subscribers, you can create a compelling upgrade path for free users while providing professional-grade tools for your paying customers.