🔔 Push Notifications Guide

Overview

PRO PDF now has a fully functional push notification system that allows users to receive real-time updates about their PDF processing tasks, even when they're not actively viewing the application.

✅ What's Configured

1. VAPID Keys Generated

VAPID (Voluntary Application Server Identification) keys enable secure push notifications:

```
NEXT_PUBLIC_VAPID_PUBLIC_KEY=BBn-
fvE_qsl7s_DZ00QnzppcGORxCZ7e4BcHdF3fqfKxGbgfw1rs4rYNOOdW-0mE97WZTHtLc1iq32wHTXIX35ec
VAPID_PRIVATE_KEY=sO4oCn-9mpeDDt4v9YBcmT9YUVDA_KylWFBM38ulecc
VAPID_SUBJECT=mailto:admin@propdf.com
```

Security:
- ✅ Public key is safe to expose in client-side code
- ⚠️ **Private key MUST remain secret** - never commit to public repositories
- ✅ Subject identifies your application to push services

2. Files Updated

Library Files:
- `lib/push-notifications.ts` - Core push notification functions
- Updated to use correct environment variables
- Added VAPID subject configuration

API Routes:
- `app/api/notifications/send/route.ts` - Send push notifications
- `app/api/notifications/subscribe/route.ts` - Subscribe/unsubscribe endpoints
- Updated to use environment variables from .env

Components:
- `components/notification-manager.tsx` - User-facing notification settings
- Already correctly configured to use NEXT_PUBLIC_VAPID_PUBLIC_KEY

Service Worker:
- `public/sw.js` - Handles push events and displays notifications
- Includes notification click handling and action buttons

🧪 How to Test Push Notifications

Step 1: Start the Development Server

```
cd /home/ubuntu/pro_pdf/nextjs_space
yarn dev
```

Open your browser to `http://localhost:3000`

Step 2: Login as Test User

```
Email: john@doe.com
Password: johndoe123
```

Step 3: Enable Notifications

1. Navigate to **Settings** page (`/settings`)
2. Click on the **Notifications** tab
3. Click the **“Enable Push Notifications”** button
4. Your browser will ask for permission - click **“Allow”**
5. You should see: ✅ “Push notifications enabled successfully!”

Step 4: Send a Test Notification

Once notifications are enabled, you’ll see a **“Send Test Notification”** button:

1. Click the **“Send Test Notification”** button
2. Within seconds, you should receive a browser notification:
   - **Title:** “Test Notification”
   - **Body:** “This is a test notification from PRO PDF!”
   - **Actions:** “View” and “Dismiss” buttons

Step 5: Test with Real Tasks

Notifications are automatically sent when:

1. **PDF Processing Completes:**
   - Go to any tool (e.g., `/tools/merge`)
   - Upload and process files
   - When complete, you’ll receive: “Task Completed Successfully!”

2. **Batch Processing Updates:**
   - Process multiple files
   - Receive progress notifications: “Processed X of Y files”

3. **Background Jobs:**
   - Long-running tasks tracked in `/jobs` page
   - Notifications sent when jobs complete or fail

🌐 Browser Compatibility

✅ Supported Browsers:

Browser	Desktop	Mobile
Chrome	✅	✅
Edge	✅	✅
Firefox	✅	✅
Safari	✅ (macOS 13+)	✅ (iOS 16.4+)
Opera	✅	✅
Samsung Internet	-	✅

❌ Not Supported:

- Safari on macOS 12 or older
- iOS Safari on iOS 16.3 or older
- Internet Explorer (all versions)

Graceful Degradation:
The app automatically detects browser support and shows appropriate messages to users on unsupported browsers.

🔒 Security & Privacy

User Control

1. **Opt-in Only:**
   - Users must explicitly enable notifications
   - Never sent without permission

2. **Easy Disable:**
   - Users can disable at any time in Settings
   - Unsubscribe removes subscription from database

3. **Browser Permissions:**
   - Respects native browser permission settings
   - If user blocks in browser, app handles gracefully

Data Storage

Database Fields Added:

```
model User {
  pushSubscription      String?  @db.Text
  notificationsEnabled  Boolean  @default(false)
}
```

- `pushSubscription`: Encrypted subscription object from browser
- `notificationsEnabled`: User preference flag

VAPID Key Security

⚠️ **IMPORTANT: Never commit private keys to public repositories!**

If you need to regenerate VAPID keys:

```
const webpush = require('web-push');
const vapidKeys = webpush.generateVAPIDKeys();
console.log('Public Key:', vapidKeys.publicKey);
console.log('Private Key:', vapidKeys.privateKey);
```

Then update your `.env` file with the new keys.

📡 Notification Types

1. Job Completion Notifications

Triggered when:
- PDF merge completes
- File conversion finishes
- Compression task done
- Any PDF tool operation completes

Example:

```
import { notifyJobCompletion } from '@/lib/push-notifications';

await notifyJobCompletion(
  'user@example.com',
  'merge',
  'job-123',
  true // success
);
```

User Receives:
- **Title:** “Task Completed Successfully!”
- **Body:** “Your merge task has finished processing.”
- **Action:** Click to view job details

2. Batch Progress Notifications

Triggered when:
- Processing multiple files

- Batch operations in progress
- Periodic progress updates

Example:

```
import { notifyBatchProgress } from '@/lib/push-notifications';

await notifyBatchProgress(
  'user@example.com',
  5, // completed
  10 // total
);
```

User Receives:
- **Title:** “Batch Processing Update”
- **Body:** “Processed 5 of 10 files”
- **Action:** Click to view dashboard

3. Custom Notifications

For any custom use case:

```
import { sendPushNotification } from '@/lib/push-notifications';

await sendPushNotification('user@example.com', {
  title: 'Custom Alert',
  body: 'Something important happened!',
  url: '/custom-page',
  jobId: 'optional-job-id',
  tag: 'custom-notification'
});
```

🛠️ Advanced Configuration

Notification Appearance

Customize in `public/sw.js`:

```
self.registration.showNotification(data.title, {
  body: data.body,
  icon: '/icon-192.png', // Your app icon
  badge: '/badge-icon.png', // Small monochrome icon
  vibrate: [200, 100, 200], // Vibration pattern
  tag: data.tag,
  requireInteraction: false, // Auto-dismiss
  actions: [
    { action: 'view', title: 'View', icon: '/view-icon.png' },
    { action: 'dismiss', title: 'Dismiss', icon: '/close-icon.png' }
  ]
});
```

Notification Click Behavior

In `public/sw.js`, customize what happens when user clicks:

```
self.addEventListener('notificationclick', (event) => {
  event.notification.close();

  if (event.action === 'view') {
    // Open specific page
    event.waitUntil(
      clients.openWindow(event.notification.data.url || '/dashboard')
    );
  }
  // 'dismiss' action automatically handled
});
```

Rate Limiting

Avoid spamming users with too many notifications:

```
// In lib/push-notifications.ts
const RATE_LIMIT = 10; // Max notifications per hour
const rateLimitKey = `notif-limit:${userEmail}`;

// Check Redis or in-memory cache
const count = await redis.incr(rateLimitKey);
if (count === 1) {
  await redis.expire(rateLimitKey, 3600); // 1 hour
}

if (count > RATE_LIMIT) {
  console.log('Rate limit exceeded for', userEmail);
  return false;
}
```

🐛 Troubleshooting

Problem: “VAPID public key not configured”

Solution:

```
# Verify .env file has the keys
cat .env | grep VAPID

# Restart dev server
yarn dev
```

Problem: “Push notifications are not supported”

Cause: Browser doesn’t support push notifications

Solution:
- Update browser to latest version
- Use Chrome, Firefox, or Edge
- On iOS, requires iOS 16.4+ with Safari

Problem: Service worker not registering

Solution:

```
# Check browser console for errors
# Verify service worker file exists
ls -la public/sw.js

# Clear browser cache and reload
# Chrome: DevTools > Application > Clear Storage > Clear site data
```

Problem: Notifications not appearing

Checklist:
1. ✅ User granted browser permission?
2. ✅ Notifications enabled in app settings?
3. ✅ Service worker registered successfully?
4. ✅ Check browser console for errors
5. ✅ Try sending test notification
6. ✅ Check browser notification settings (OS level)

Debug Mode:

```
// In components/notification-manager.tsx
console.log('Permission:', Notification.permission);
console.log('ServiceWorker:', 'serviceWorker' in navigator);
console.log('PushManager:', 'PushManager' in window);
```

Problem: Notification subscription expired (410 error)

Cause: Browser invalidated the push subscription

Automatic Fix: App automatically removes expired subscriptions

User Action: Re-enable notifications in Settings

📊 Monitoring & Analytics

Track Notification Performance

Add logging to measure:

1. **Subscription Rate:** % of users who enable notifications
2. **Click-through Rate:** % of notifications clicked
3. **Opt-out Rate:** % of users who disable
4. **Delivery Success Rate:** % of notifications sent vs. failed

Example:

```
// In lib/push-notifications.ts
await sendPushNotification(userEmail, payload);

// Log to analytics
await analytics.track('push_notification_sent', {
  userId: userEmail,
  type: payload.tag,
  timestamp: new Date()
});
```

🚀 Production Deployment

1. Environment Variables

Ensure your production environment has:

```
NEXT_PUBLIC_VAPID_PUBLIC_KEY=your_public_key_here
VAPID_PRIVATE_KEY=your_private_key_here
VAPID_SUBJECT=mailto:your-support-email@domain.com
```

2. HTTPS Requirement

⚠️ **Push notifications require HTTPS in production**

- Development: Works on localhost without HTTPS
- Production: MUST use HTTPS (https://yourdomain.com)

3. Service Worker Path

Ensure service worker is accessible:

```
// In app/layout.tsx
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/sw.js')
    .then(reg => console.log('SW registered:', reg))
    .catch(err => console.error('SW registration failed:', err));
}
```

4. CDN Configuration

If using a CDN (like Cloudflare):

```
# Don't cache service worker
location = /sw.js {
  add_header Cache-Control "no-cache, no-store, must-revalidate";
}
```

📈 Future Enhancements

Planned Features:

1. **Notification Preferences:**
   - Allow users to choose which notifications to receive
   - Quiet hours scheduling
   - Notification frequency controls

2. **Rich Notifications:**
   - Inline images (PDF thumbnails)
   - Progress bars for batch operations
   - Multiple action buttons

3. **Notification History:**
   - View past notifications
   - Re-trigger actions from old notifications
   - Notification analytics dashboard

4. **Multi-device Support:**
   - Sync notification preferences across devices
   - Device-specific notification settings
   - "Send to this device" option

5. **Advanced Features:**
   - Background sync for offline operations
   - Periodic background sync
   - Notification grouping/stacking

📚 Additional Resources

Documentation:

- Web Push API (https://developer.mozilla.org/en-US/docs/Web/API/Push_API)
- Service Worker API (https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API)
- Notifications API (https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API)
- VAPID Protocol (https://tools.ietf.org/html/rfc8292)

Libraries:

- web-push (npm) (https://www.npmjs.com/package/web-push)
- Web Push Book (https://web-push-book.gauntface.com/)

Testing Tools:

- Push Notification Tester (https://tests.peter.sh/notification-generator/)
- Chrome DevTools > Application > Service Workers
- Firefox DevTools > Storage > Service Workers

✅ Summary

What's Working:
- ✅ VAPID keys generated and configured
- ✅ Push notification system fully functional
- ✅ User can enable/disable notifications in Settings
- ✅ Test notification button available
- ✅ Automatic notifications for job completion
- ✅ Batch progress notifications
- ✅ Service worker handles push events
- ✅ Notification click handling with actions
- ✅ Browser compatibility detection
- ✅ Graceful degradation for unsupported browsers
- ✅ Secure VAPID configuration
- ✅ Database integration for subscriptions

Status: 🎉 Production Ready!

Your push notification system is fully configured and ready to use. Users can now receive real-time updates about their PDF processing tasks, even when they're not actively viewing the application.

Generated: December 2, 2025
Version: 2.1
PRO PDF - Professional PDF Tools