🚀 Guide d'Installation - Timer Journalier

📋 Prérequis

Votre serveur Ubuntu doit avoir :

- Ubuntu 20.04 ou supérieur
- Accès sudo
- Connexion internet

1️⃣ Installation de Node.js

```bash
# Mettre à jour le système
sudo apt update
sudo apt upgrade -y

# Installer Node.js (version 18 LTS)
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install -y nodejs

# Vérifier l'installation
node --version
npm --version
```

2️⃣ Installation de SQLite

```bash
# Installer SQLite3
sudo apt install sqlite3

# Vérifier l'installation
sqlite3 --version
```

3️⃣ Créer la structure du projet

```bash
# Créer le dossier du projet
mkdir -p /var/www/timer-app
cd /var/www/timer-app

# Créer les sous-dossiers
mkdir public
```

Structure finale :

```
/var/www/timer-app/
├── server.js          # Serveur backend
├── package.json       # Dépendances
├── schema.sql         # Schéma de base de données
├── timer_app.db       # Base de données (sera créée)
└── public/            # Fichiers HTML/CSS/JS
    ├── index.html     # Page employé (timer)
    ├── admin.html     # Page admin
    └── login.html     # Page de connexion (à créer)
```

4️⃣ Créer les fichiers

Fichier 1 : server.js

Copiez le contenu du fichier `server.js` que je vous ai donné.

Fichier 2 : package.json

Copiez le contenu du fichier `package.json`.

Fichier 3 : schema.sql

Copiez le contenu du fichier `schema.sql` (base de données).

5️⃣ Créer la base de données

```bash
cd /var/www/timer-app

# Créer la base de données
sqlite3 timer_app.db < schema.sql

# Vérifier que les tables existent
sqlite3 timer_app.db "SELECT name FROM sqlite_master WHERE type='table';"

# Vérifier l'admin par défaut
sqlite3 timer_app.db "SELECT email, role FROM users WHERE role='admin';"
```

Résultat attendu :

- `admin@entreprise.fr` avec le rôle `admin`

6️⃣ Installer les dépendances Node.js

```bash
cd /var/www/timer-app
npm install
```

Cela installe :

- Express (serveur web)
- SQLite3 (base de données)
- bcrypt (hash des mots de passe)
- jsonwebtoken (authentification)
- cors (sécurité)

7️⃣ Placer vos fichiers HTML

```bash
# Copier vos pages dans le dossier public/
cp /chemin/vers/timer.html /var/www/timer-app/public/index.html
cp /chemin/vers/admin.html /var/www/timer-app/public/admin.html
```

8️⃣ Démarrer le serveur

Mode développement (avec auto-reload) :

```bash
npm run dev
```

Mode production :

```bash
npm start
```

Avec PM2 (recommandé pour production) :

```bash
# Installer PM2 globalement
sudo npm install -g pm2

# Démarrer l'application
pm2 start server.js --name timer-app

# Sauvegarder la configuration
pm2 save

# Démarrage automatique au boot
pm2 startup
# Suivre les instructions affichées
```

9️⃣ Tester l'installation

Test 1 : Vérifier que le serveur tourne

```bash
curl http://localhost:3000
```

Test 2 : Tester la connexion admin

```bash
curl -X POST http://localhost:3000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email":"admin@entreprise.fr","password":"admin123"}'
```

Résultat attendu : Un token JWT

🔟 Configuration du pare-feu

```bash
# Autoriser le port 3000
sudo ufw allow 3000

# Vérifier le statut
sudo ufw status
```

🌐 Accéder à l'application

En local sur le serveur :

```
http://localhost:3000
```

Depuis un autre ordinateur :

```
http://IP_DU_SERVEUR:3000
```

Par exemple : `http://192.168.1.100:3000`

🔧 Commandes utiles

Voir les logs PM2 :

```bash
pm2 logs timer-app
```

Redémarrer l'application :

```bash
pm2 restart timer-app
```

Arrêter l'application :

```bash
pm2 stop timer-app
```

Vérifier le statut :

```bash
pm2 status
```

Inspecter la base de données :

```bash
sqlite3 timer_app.db
# Puis dans sqlite :
.tables              # Liste des tables
SELECT * FROM users;     # Voir les utilisateurs
.quit                # Quitter
```

📊 Comptes par défaut

Administrateur :

- **Email :** `admin@entreprise.fr`
- **Mot de passe :** `admin123`

Employés de test :

- **Jean Dupont :** `jean.dupont@entreprise.fr` / `password123`
- **Marie Martin :** `marie.martin@entreprise.fr` / `password123`
- **Pierre Dubois :** `pierre.dubois@entreprise.fr` / `password123`

⚠️ **Important :** Changez ces mots de passe en production !

🔒 Sécurité - Production

Changer le JWT_SECRET :

Dans `server.js`, ligne 9 :

```javascript
const JWT_SECRET = 'VOTRE_SECRET_COMPLEXE_ICI';
```

Utiliser HTTPS :

Installez un certificat SSL avec Let's Encrypt ou placez derrière un reverse proxy (nginx).

Changer les mots de passe par défaut :

```bash
sqlite3 timer_app.db
UPDATE users SET password_hash = 'NOUVEAU_HASH' WHERE email = 'admin@entreprise.fr';
```

📦 Sauvegarde

Sauvegarder la base de données :

```bash
# Créer une sauvegarde
cp timer_app.db timer_app_backup_$(date +%Y%m%d).db

# Ou via cron (automatique chaque jour à 2h)
crontab -e
# Ajouter : 0 2 * * * cp /var/www/timer-app/timer_app.db /var/backups/timer_$(date +\%Y\%m\%d).db
```

❓ Dépannage

Le serveur ne démarre pas :

```bash
# Vérifier les logs
npm start
# Lire les erreurs affichées
```

Port déjà utilisé :

```bash
# Changer le port dans server.js ligne 8
const PORT = 3001; // Au lieu de 3000
```

Erreur de permissions :

```bash
# Donner les bonnes permissions
sudo chown -R $USER:$USER /var/www/timer-app
chmod -R 755 /var/www/timer-app
```

🎉 C'est terminé !

Votre application est maintenant installée et fonctionnelle !

Prochaines étapes :

1. Créer la page de connexion
2. Connecter les pages HTML aux API
3. Tester avec de vrais utilisateurs

Besoin d'aide ? Demandez-moi ! 🚀