Titre : **PrintPay**

**1. Définition des fonctionnalités clés**

Avant d’écrire du code, il faut définir ce que ton application doit faire.

**Fonctionnalités principales :**

✅ **Gestion des utilisateurs** (enseignants)  
✅ **Suivi des impressions et photocopies** (nombre, type, date)  
✅ **Gestion des paiements** (suivi des montants payés et dus)  
✅ **Génération de rapports** (historique des transactions)  
✅ **Notifications** (rappels de paiement, confirmation d’impression)

**2. Choix des technologies**

📱 **Frontend (App mobile) :** React Native + TypeScript (ou JavaScript)  
🛠️ **UI/UX :** React Native Paper ou Native Base pour un design moderne  
💾 **Backend :** Firebase (facile à intégrer avec React Native) ou Node.js + Express  
📊 **Base de données :** Firebase Firestore ou PostgreSQL/MySQL avec Supabase  
🔐 **Authentification :** Firebase Auth ou Auth0

**3. Développement de l'application**

**A. Mise en place du projet React Native**

1️⃣ Installer **Node.js** et **React Native CLI** (ou Expo pour une configuration plus simple)  
2️⃣ Créer un projet React Native :

npx react-native init GestionImpressions # Avec React Native CLI

# ou

npx create-expo-app GestionImpressions # Avec Expo (plus simple)

3️⃣ Lancer l’application sur un émulateur ou un téléphone :

cd GestionImpressions

npx expo start # ou react-native run-android / run-ios

**B. Création des interfaces (UI/UX)**

* **Écran d’accueil** (connexion et inscription)
* **Dashboard** (aperçu des impressions et paiements)
* **Écran de gestion des impressions** (ajouter, modifier, supprimer)
* **Écran de paiement** (voir les transactions, enregistrer les paiements)

**C. Gestion des données avec Firebase**

1️⃣ **Installer Firebase SDK**

npm install firebase

2️⃣ **Configurer Firebase dans l’app**  
3️⃣ **Créer des collections Firestore** :

* **Users** (enseignants)
* **Impressions** (date, nombre, type)
* **Paiements** (montant, date)

**D. Ajouter l’authentification (Google, email/password)**

1️⃣ **Activer Firebase Auth**  
2️⃣ **Ajouter l’inscription et connexion**

**4. Tests et Déploiement**

✅ Tester sur **Android & iOS**  
✅ Déployer avec **Expo** ou sur le **Play Store/App Store**

**Résumé du projet en étapes simples :**

1️⃣ **Définir les fonctionnalités**  
2️⃣ **Créer l’interface en React Native**  
3️⃣ **Gérer les données avec Firebase**  
4️⃣ **Ajouter les paiements et l’historique**  
5️⃣ **Tester et publier l’application**

Voici un plan étape par étape pour créer cette application :

**Étape 1** : Configuration de l'Environnement de Développement

**1. Installer Node.js et npm**

**2. Installer React Native CLI**

bash

npm install -g react-native-cli

**3. Initialiser le Projet**

bash

npx react-native init PrintPay

**Étape 2** : Structure du Projet

Organiser le projet en différentes sections :

PrintPay/

├── android/

├── ios/

├── src/

│ ├── components/

│ ├── screens/

│ ├── services/

│ └── assets/

└── App.js

**Étape 3 :** Créer les Écrans et Composants

**1. Page de Connexion**

javascript

// src/screens/LoginScreen.js

import React, { useState } from 'react';

import { View, TextInput, Button, Text } from 'react-native';

const LoginScreen = ({ navigation }) => {

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const handleLogin = () => {

// Implémenter la logique de connexion

navigation.navigate('Dashboard');

};

return (

<View>

<Text>Email</Text>

<TextInput value={email} onChangeText={setEmail} />

<Text>Password</Text>

<TextInput value={password} onChangeText={setPassword} secureTextEntry />

<Button title="Login" onPress={handleLogin} />

</View>

);

};

export default LoginScreen;

**2. Tableau de Bord (Dashboard)**

javascript

// src/screens/Dashboard.js

import React from 'react';

import { View, Button } from 'react-native';

const Dashboard = ({ navigation }) => {

return (

<View>

<Button title="Add Teacher" onPress={() => navigation.navigate('AddTeacher')} />

<Button title="Manage Transactions" onPress={() => navigation.navigate('ManageTransactions')} />

</View>

);

};

export default Dashboard;

**3. Ajouter un Enseignant**

javascript

// src/screens/AddTeacher.js

import React, { useState } from 'react';

import { View, TextInput, Button } from 'react-native';

const AddTeacher = ({ navigation }) => {

const [name, setName] = useState('');

const handleAddTeacher = () => {

// Implémenter la logique d'ajout d'un enseignant

navigation.goBack();

};

return (

<View>

<Text>Teacher Name</Text>

<TextInput value={name} onChangeText={setName} />

<Button title="Add Teacher" onPress={handleAddTeacher} />

</View>

);

};

export default AddTeacher;

**4. Gérer les Transactions**

javascript

// src/screens/ManageTransactions.js

import React, { useState } from 'react';

import { View, Text, TextInput, Button } from 'react-native';

const ManageTransactions = () => {

const [teacherName, setTeacherName] = useState('');

const [copies, setCopies] = useState('');

const [copiesDate, setCopiesDate] = useState('');

const [prints, setPrints] = useState('');

const [printsDate, setPrintsDate] = useState('');

const handleAddTransaction = () => {

// Implémenter la logique d'ajout de transactions

};

return (

<View>

<Text>Teacher Name</Text>

<TextInput value={teacherName} onChangeText={setTeacherName} />

<Text>Copies</Text>

<TextInput value={copies} onChangeText={setCopies} keyboardType="numeric" />

<Text>Copies Date</Text>

<TextInput value={copiesDate} onChangeText={setCopiesDate} />

<Text>Prints</Text>

<TextInput value={prints} onChangeText={setPrints} keyboardType="numeric" />

<Text>Prints Date</Text>

<TextInput value={printsDate} onChangeText={setPrintsDate} />

<Button title="Add Transaction" onPress={handleAddTransaction} />

</View>

);

};

export default ManageTransactions;

**Étape 4** : Navigation et Routage

Utiliser react-navigation pour gérer la navigation entre les différents écrans.

bash

npm install @react-navigation/native @react-navigation/stack

Configurer la navigation dans le fichier App.js :

```javascript

import 'react-native-gesture-handler';

import \* as React from 'react';

import { NavigationContainer } from '@react-navigation/native';

import { createStackNavigator } from '@react-navigation/stack';

import LoginScreen from './src/screens/LoginScreen';

import Dashboard from './src/screens/Dashboard';

import AddTeacher from './src/screens/AddTeacher';

import ManageTransactions from './src/screens/ManageTransactions';

const Stack = createStackNavigator();

const App = () => {

return (

<NavigationContainer>

<Stack.Navigator initialRouteName="Login">

<Stack.Screen name="Login" component={LoginScreen} />

<Stack.Screen name="Dashboard" component={Dashboard} />

<Stack.Screen name="AddTeacher" component={AddTeacher} />

<Stack.Screen name="ManageTransactions" component={ManageTransactions} />

</Stack.Navigator>

</NavigationContainer>

);

};

export default App;

### **Étape 5** : Gestion des Données

Utiliser une base de données comme Firebase Firestore pour stocker les enseignants et les transactions :

bash

npm install @react-native-firebase/firestore

Exemple d'intégration avec Firestore pour ajouter un enseignant :

javascript

import firestore from '@react-native-firebase/firestore';

// Fonction pour ajouter un enseignant

const addTeacher = async (teacherName) => {

await firestore().collection('teachers').add({

name: teacherName,

});

};

```

**Étape 6** : Test et Débogage

- Effectuer des tests unitaires et d'intégration pour s'assurer que toutes les fonctionnalités fonctionnent correctement.

- Utiliser les outils de débogage de React Native pour résoudre les problèmes.