Dream Journal — Full Stack (FastAPI + React)

A step-by-step guide to build the **Dream Journal** web app frontend with React and Tailwind CSS. Users can register/login, post dreams with images, like/unlike, comment, and view others' dreams. This explanation focuses on helping you understand the **frontend structure and flow**, especially if you are new to React.

🖥️Frontend Overview

We'll build a modern single-page application (SPA) using **React** (created via Vite for speed) and **Tailwind CSS** for styling. The frontend will connect to the FastAPI backend via REST APIs.

📦Folder Structure

```
frontend/
├── index.html              # Root HTML file
├── package.json            # Project dependencies and scripts
├── vite.config.ts          # Vite configuration
├── tailwind.config.cjs     # Tailwind setup
├── postcss.config.cjs      # PostCSS setup
└── src/
    ├── main.tsx            # Application entry point
    ├── App.tsx             # Main layout and routes
    ├── api/
    │   └── client.ts       # Axios configuration
    ├── store/
    │   └── auth.ts         # Authentication helper functions
    ├── components/
    │   └── DreamCard.tsx   # Reusable dream display component
    ├── pages/
    │   ├── Home.tsx        # Home page - list of dreams
    │   ├── DreamDetail.tsx # Detailed dream view with comments
    │   ├── Login.tsx       # Login form
    │   ├── Register.tsx    # Registration form
    │   ├── CreateDream.tsx # Create dream page
    │   └── Profile.tsx     # User profile and dreams
    └── styles.css          # Tailwind imports
```

🧱Step 1: Create the React Project

Run the following commands to create the frontend:

```
npm create vite@latest dream-journal-frontend -- --template react-ts
cd dream-journal-frontend
npm install
```

Then install Tailwind CSS:

```
npm install -D tailwindcss postcss autoprefixer
npx tailwindcss init -p
```

Update **tailwind.config.cjs**:

```
module.exports = {
  content: ['./index.html', './src/**/*.{js,ts,jsx,tsx}'],
  theme: { extend: {} },
  plugins: [],
}
```

Create **src/styles.css**:

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

🧰Step 2: Configure API Client (Axios)

Create **src/api/client.ts**:

```
import axios from 'axios'

export const API = axios.create({
  baseURL: 'http://localhost:8000/api/v1', // backend endpoint
})

API.interceptors.request.use(config => {
  const token = localStorage.getItem('token')
  if (token) config.headers.Authorization = `Bearer ${token}`
  return config
})
```

This ensures every request includes the JWT token if logged in.

🔐Step 3: Handle Authentication

Create **src/store/auth.ts**:

```
export const auth = {
  isAuthed() { return !!localStorage.getItem('token') },
  set(token: string) { localStorage.setItem('token', token) },
  clear() { localStorage.removeItem('token') }
}
```

This stores the JWT token in the browser, checks if logged in, and clears on logout.

🧭Step 4: Set Up Routing and Main Layout

We use **React Router** to handle navigation between pages.

Install it:

```
npm install react-router-dom
```

Create **src/main.tsx**:

```
import React from 'react'
import { createRoot } from 'react-dom/client'
import { BrowserRouter, Routes, Route, Navigate } from 'react-router-dom'
import './styles.css'
import App from './App'
import Home from './pages/Home'
import DreamDetail from './pages/DreamDetail'
import Login from './pages/Login'
import Register from './pages/Register'
import CreateDream from './pages/CreateDream'
import Profile from './pages/Profile'
import { auth } from './store/auth'

function PrivateRoute({ children }: { children: JSX.Element }) {
  return auth.isAuthed() ? children : <Navigate to="/login" />
}

createRoot(document.getElementById('root')!).render(
  <BrowserRouter>
    <Routes>
      <Route element={<App />}>
        <Route path="/" element={<Home />} />
        <Route path="/dream/:id" element={<DreamDetail />} />
        <Route path="/login" element={<Login />} />
        <Route path="/register" element={<Register />} />
        <Route path="/create" element={<PrivateRoute><CreateDream /></
PrivateRoute>} />
        <Route path="/me" element={<PrivateRoute><Profile /></
PrivateRoute>} />
      </Route>
    </Routes>
  </BrowserRouter>
)
```

The `PrivateRoute` ensures protected pages require login.

🪠Step 5: Create Navigation (App.tsx)

This file defines the common navigation and layout.

```
import { Outlet, Link, useNavigate } from 'react-router-dom'
import { auth } from './store/auth'

export default function App() {
  const nav = useNavigate()
  return (
    <div className="max-w-4xl mx-auto p-4">
      <nav className="flex items-center justify-between py-3 border-b">
        <Link to="/" className="font-bold text-xl">🌙 Dream Journal</Link>
        <div className="space-x-3">
          <Link to="/">Home</Link>
          {auth.isAuthed() && <Link to="/create">New Dream</Link>}
          {auth.isAuthed() && <Link to="/me">Profile</Link>}
          {!auth.isAuthed() && <Link to="/login">Login</Link>}
          {!auth.isAuthed() && <Link to="/register">Register</Link>}
          {auth.isAuthed() && <button className="text-red-600" onClick={()
=> {auth.clear(); nav('/')}}>Logout</button>}
        </div>
      </nav>
      <Outlet /> {/* where pages will render */}
    </div>
  )
}
```

💸Step 6: Home Page (List All Dreams)

Fetches all dreams and displays them with like buttons.

```
import { useEffect, useState } from 'react'
import { API } from '../api/client'
import DreamCard from '../components/DreamCard'

export default function Home() {
  const [dreams, setDreams] = useState<any[]>([])

  const fetchAll = async () => {
    const { data } = await API.get('/dreams/')
    setDreams(data)
  }

  useEffect(() => { fetchAll() }, [])

  const onLike = async (id:number) => {
    await API.post(`/likes/${id}`)
    fetchAll()
  }

  return (
    <div>
      <h2 className="text-2xl font-bold mb-3">Latest Dreams</h2>
      {dreams.map(d => <DreamCard key={d.id} dream={d} onLike={onLike} />)}
    </div>
  )
}
```

🕖Step 7: DreamCard Component

A reusable UI block for displaying a single dream.

```
import { Link } from 'react-router-dom'

export default function DreamCard({ dream, onLike }: { dream: any, onLike?:
(id:number)=>void }) {
  return (
    <div className="bg-white rounded-lg p-4 shadow mb-4">
      <div className="flex items-center gap-2 text-sm text-gray-500">
        <span>@{dream.author.username}</span>
        <span>•</span>
        <span>Likes: {dream.like_count}</span>
      </div>
      <Link to={`/dream/${dream.id}`}>
        <h3 className="text-lg font-semibold mt-1">{dream.title}</h3>
        <p className="text-gray-700 mt-1 line-clamp-3">{dream.body}</p>
        {dream.image_path && <img className="mt-2 rounded" src={`http://
localhost:8000/${dream.image_path}`} />}
```

```
      </Link>
      {onLike && <button className="mt-2 text-blue-600" onClick={() =>
onLike(dream.id)}>Like/Unlike</button>}
    </div>
  )
}
```

Step 8: Login and Register Pages

Handle authentication forms.

`Login.tsx`

```
import { useState } from 'react'
import { API } from '../api/client'
import { useNavigate } from 'react-router-dom'
import { auth } from '../store/auth'

export default function Login() {
  const [email, setEmail] = useState('')
  const [password, setPassword] = useState('')
  const nav = useNavigate()

  const submit = async () => {
    const { data } = await API.post('/auth/login', { email, password })
    auth.set(data.access_token)
    nav('/')
  }

  return (
    <div className="max-w-sm mx-auto bg-white p-4 rounded shadow">
      <h2 className="text-xl font-bold mb-2">Login</h2>
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Email" value={email} onChange={e=>setEmail(e.target.value)} />
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Password" type="password" value={password}
onChange={e=>setPassword(e.target.value)} />
      <button className="bg-blue-600 text-white w-full py-2 rounded"
onClick={submit}>Login</button>
    </div>
  )
}
```

`Register.tsx`

```
import { useState } from 'react'
import { API } from '../api/client'
import { useNavigate } from 'react-router-dom'

export default function Register() {
  const [email, setEmail] = useState('')
  const [username, setUsername] = useState('')
  const [password, setPassword] = useState('')
  const nav = useNavigate()

  const submit = async () => {
    await API.post('/auth/register', { email, username, password })
    nav('/login')
  }

  return (
    <div className="max-w-sm mx-auto bg-white p-4 rounded shadow">
      <h2 className="text-xl font-bold mb-2">Register</h2>
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Email" value={email} onChange={e=>setEmail(e.target.value)} />
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Username" value={username}
onChange={e=>setUsername(e.target.value)} />
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Password" type="password" value={password}
onChange={e=>setPassword(e.target.value)} />
      <button className="bg-green-600 text-white w-full py-2 rounded"
onClick={submit}>Create Account</button>
    </div>
  )
}
```

🖋️Step 9: Create Dream Page

Allows logged-in users to submit dreams.

```
import { useState } from 'react'
import { API } from '../api/client'
import { useNavigate } from 'react-router-dom'

export default function CreateDream() {
  const [title, setTitle] = useState('')
  const [body, setBody] = useState('')
  const [tags, setTags] = useState('')
  const [file, setFile] = useState<File | null>(null)
  const nav = useNavigate()

  const submit = async () => {
    const form = new FormData()
    form.append('title', title)
    form.append('body', body)
    form.append('tags', tags)
    if (file) form.append('image', file)
    await API.post('/dreams/', form, { headers: { 'Content-Type': 'multipart/
form-data' } })
    nav('/')
  }

  return (
    <div className="max-w-xl mx-auto bg-white p-4 rounded shadow">
      <h2 className="text-xl font-bold mb-2">Share a Dream</h2>
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Title" value={title} onChange={e=>setTitle(e.target.value)} />
      <textarea className="border rounded w-full px-2 py-1 mb-2 h-40"
placeholder="Describe your dream..." value={body}
onChange={e=>setBody(e.target.value)} />
      <input className="border rounded w-full px-2 py-1 mb-2"
placeholder="Tags (comma separated)" value={tags}
onChange={e=>setTags(e.target.value)} />
      <input type="file" onChange={e=>setFile(e.target.files?.[0] || null)}
className="mb-2" />
      <button className="bg-indigo-600 text-white px-4 py-2 rounded"
onClick={submit}>Publish</button>
    </div>
  )
}
```

Step 10: Profile Page

Shows logged-in user info and their posted dreams.

```
import { useEffect, useState } from 'react'
import { API } from '../api/client'

export default function Profile() {
  const [me, setMe] = useState<any>(null)
  const [myDreams, setMyDreams] = useState<any[]>([])

  useEffect(() => {
    (async () => {
      const { data } = await API.get('/auth/me')
      setMe(data)
      const { data: list } = await API.get('/dreams/')
      setMyDreams(list.filter((d:any)=>d.author.id===data.id))
    })()
  }, [])

  if (!me) return <div>Loading...</div>
  return (
    <div>
      <h2 className="text-2xl font-bold mb-2">@{me.username}</h2>
      <p className="text-gray-600">{me.email}</p>
      <h3 className="text-xl font-semibold mt-4 mb-2">My Dreams</h3>
      {myDreams.map(d => (
        <div key={d.id} className="bg-white rounded p-3 shadow mb-2">
          <div className="font-semibold">{d.title}</div>
          <div className="text-sm text-gray-600">Likes: {d.like_count}</div>
        </div>
      ))}
    </div>
  )
}
```

🗜️ Step 11: Run the Frontend

Start the development server:

```
npm run dev
```

Then open **http://localhost:5173** in your browser.

You'll see a beautiful interface where you can register, log in, and start sharing dreams.

Tips for Beginners

- **JSX:** It's HTML mixed with JavaScript — used to describe what UI should look like.
- **State:** Variables managed by React; when updated, UI re-renders.
- **Props:** Data passed from parent components (e.g., Dream list → DreamCard).
- **Hooks:** Functions like `useState` and `useEffect` for managing state and side effects.
- **Tailwind:** A utility-first CSS framework — apply classes directly like `bg-blue-600 text-white`.

By following these steps, you'll not only get a working app but also a clear understanding of **how React apps are structured and communicate with an API**.