ChatGPT

Admin Seed Page with Modern UI and Drizzle ORM

Backend Setup

1. `users` Table (Drizzle Schema)

```
import { pgTable, serial, varchar, text } from 'drizzle-orm/pg-core';

export const users = pgTable('users', {
  id: serial('id').primaryKey(),
  email: varchar('email', { length: 255 }).notNull().unique(),
  password: text('password').notNull(),
  role: varchar('role', { length: 50 }).default('user')
});
```

2. Password Hashing Utility

```
// src/lib/server/utils/hash.ts
import bcrypt from 'bcryptjs';

export async function hashPassword(plain: string) {
  const salt = await bcrypt.genSalt(10);
  return bcrypt.hash(plain, salt);
}
```

3. Seed Logic Utility

```
// src/lib/server/utils/seed-users.ts
import { db } from '../db';
import { users } from '../db/schema';
import { hashPassword } from './hash';

export async function seedDefaultUsers() {
  const existing = await
db.select().from(users).where(users.email.eq('admin@example.com'));
  if (existing.length > 0) return { message: 'Already seeded' };

  await db.insert(users).values([
    {
      email: 'admin@example.com',
```

```
      password: await hashPassword('admin123'),
      role: 'admin'
    },
    {
      email: 'user@example.com',
      password: await hashPassword('user123'),
      role: 'user'
    }
  ]);

  return { message: 'Seeded successfully' };
}
```

4. Route Handlers

`+page.server.ts`

```
import { db } from '$lib/server/db';
import { users } from '$lib/server/db/schema';
import { redirect } from '@sveltejs/kit';

export const load = async ({ locals }) => {
  if (!locals.session?.user || locals.session.user.role !== 'admin') {
    throw redirect(302, '/login');
  }

  const allUsers = await db.select().from(users);
  return { users: allUsers };
};
```

`+server.ts`

```
import { json } from '@sveltejs/kit';
import { seedDefaultUsers } from '$lib/server/utils/seed-users';

export const POST = async ({ locals }) => {
  if (!locals.session?.user || locals.session.user.role !== 'admin') {
    return json({ message: 'Unauthorized' }, { status: 401 });
  }
  const result = await seedDefaultUsers();
  return json(result);
};
```

5. `+page.svelte`

```
<script lang="ts">
  import { invalidate } from '$app/navigation';
  export let users: { id: number; email: string; role: string }[];
  let message = '';

  async function seed() {
    const res = await fetch('/admin/seed-user/seed', { method: 'POST' });
    const data = await res.json();
    message = data.message;
    await invalidate();
  }
</script>

<h1>Admin Seeder</h1>
<button on:click={seed}>Seed Users</button>
{#if message}<p>{message}</p>{/if}
<ul>
  {#each users as u}<li>{u.email} – {u.role}</li>{/each}
</ul>
```

UI Implementation Plan

Components

1. `+Sidebar.svelte` – nav with active link styling
2. `+Header.svelte` – title, search, user icon
3. `+SearchInput.svelte` – reusable search bar
4. `+FileUploadSection.svelte` – file inputs + dropdowns
5. `+AutomateUploadSection.svelte` – form inputs
6. `+AddNotesSection.svelte` – notes input
7. `+Dropdown.svelte`, `+Checkbox.svelte` – reusable elements

Global Styling

- `app.css`: 2-column layout, card UI, inputs
- Scoped styles in each component

Routing

- `src/routes/interactive-canvas/+page.svelte` to compose main layout

Component Tree

```
graph TD
    A[src/routes/interactive-canvas/+page.svelte] --> B[+Header.svelte]
    A --> C[+Sidebar.svelte]
    A --> D[Main Content Area]
    D --> E[+FileUploadSection.svelte]
    D --> F[+AutomateUploadSection.svelte]
    D --> G[+AddNotesSection.svelte]
    E --> H[+Dropdown.svelte]
    E --> I[+Checkbox.svelte]
    F --> H
    F --> I
    G --> H
    G --> I
    B --> J[+SearchInput.svelte]
```

Let me know when you're ready to scaffold the actual components and routes.