



សាកលវិទ្យាល័យនំត្បូន
NORTON UNIVERSITY



User Registration System

2025 – 2026

Y3 – DCS – NU



User Registration

Using Flask and

SQLite3

By: SEK SOCHEAT

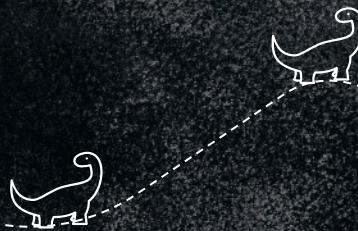
Advisor to DCS and Lecturer

Mobile: 017 879 967

Email: socheatsek@norton-u.com

socheat.sek@gmail.com

1 — Users Registration in **models**/



```

FLASK_RBAC_SQLITE3
  .venvUsers
  app
    forms
      __init__.py
      user_forms.py
    models
      __init__.py
      user.py
    routes
      __init__.py
      user_routes.py
  services
    __init__.py
    user_service.py
  static
    css
    images
    js
      main.js
  templates
    layouts
      base.html
    permissions
      roles
      users
        _form.html
        create.html
        delete_confirm.html
        detail.html
        edit.html
        index.html
    __init__.py
  instance
    users.db
  migrations
  tests
  utils
    config.py
    extensions.py
  requirements.txt
  run.py

```

config.py > ...

```

1 import os
2
3 BASE_DIR = os.path.abspath(os.path.dirname(__file__))
4
5
6 class Config:
7     SECRET_KEY = os.environ.get("SECRET_KEY", "dev-secret-key-change-me")
8     SQLALCHEMY_DATABASE_URI = (
9         os.environ.get("DATABASE_URL")
10        or "sqlite://" + os.path.join(BASE_DIR, "instance\\users.db")
11    )
12     SQLALCHEMY_TRACK_MODIFICATIONS = False
13

```

extensions.py > ...

```

1 from flask_sqlalchemy import SQLAlchemy
2 from flask_wtf import CSRFProtect
3
4 db = SQLAlchemy()
5 csrf = CSRFProtect()
6

```

requirements.txt

```

1 Flask==2.3.3
2 Flask-WTF==1.1.1
3 WTForms==3.1.2
4 email-validator==2.1.0.post1
5 Werkzeug==2.3.7
6 Flask-SQLAlchemy==3.1.1
7

```

Setting Up a Virtual Environment in Windows via Git-Bash

- Create new venv:** python -m venv .venvUser
- Activate venv via Git-Bash:** source .venvUser/scripts/activate
- Upgrade pip:** python -m pip install --upgrade pip
- Installing requirements:** pip install -r requirements.txt

EXPLORER

FLASK_RBAC_SQLITE3 .venvUsers app forms _init_.py user_forms.py models _init_.py user.py routes _init_.py user_routes.py services _init_.py user_service.py static css images js main.js templates layouts base.html permissions roles users _form.html create.html delete_confirm.html detail.html edit.html index.html _init_.py instance users.db migrations tests utils config.py extensions.py requirements.txt run.py

app > models >  _init_.py

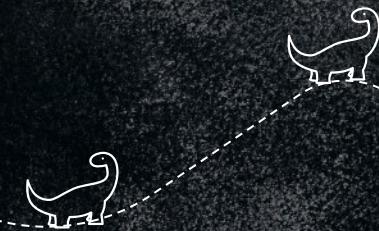
```
1 from .user import User
```

1 from datetime import datetime
2 from werkzeug.security import generate_password_hash, check_password_hash
3
4 from extensions import db
5
6
7 class User(db.Model):
8 __tablename__ = "users"
9
10 id = db.Column(db.Integer, primary_key=True)
11 username = db.Column(db.String(80), unique=True, nullable=False)
12 email = db.Column(db.String(120), unique=True, nullable=False)
13 full_name = db.Column(db.String(120), nullable=False)
14 is_active = db.Column(db.Boolean, default=True, nullable=False)
15
16 # NEW: store only the hash, never plain text
17 password_hash = db.Column(db.String(255), nullable=False)
18
19 created_at = db.Column(db.DateTime, default=datetime.utcnow, nullable=False)
20 updated_at = db.Column(
21 db.DateTime, default=datetime.utcnow, onupdate=datetime.utcnow, nullable=False
22)
23
24 # convenience helpers
25 def set_password(self, password: str) -> None:
26 self.password_hash = generate_password_hash(password)
27
28 def check_password(self, password: str) -> bool:
29 return check_password_hash(self.password_hash, password)
30
31 def __repr__(self) -> str:
32 return f"<User {self.username}>"

4

NU
INDIANA UNIVERSITY

2 — Users Registration in **templates/**



FLASK_RBAC_SQLITE3 .venvUsers app forms _init_.py user_forms.py models _init_.py user.py routes _init_.py user_routes.py services _init_.py user_service.py static css images js main.js templates layouts base.html permissions roles users _form.html create.html delete_confirm.html detail.html edit.html index.html _init_.py instance users.db migrations tests utils config.py extensions.py requirements.txt run.py

app > templates > layouts > base.html

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4      <meta charset="utf-8" />
5      <title>{% block title %}User Management{% endblock %}</title>
6      <meta name="viewport" content="width=device-width, initial-scale=1" />
7
8      <!-- Bootstrap 5 CSS -->
9      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" />
10
11     <!-- Custom CSS -->
12     <link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}" />
13     {% block extra_css %}{% endblock %}
14 </head>
15
16 <body>
17     <nav class="navbar navbar-expand-lg navbar-dark bg-primary mb-4">
18         <div class="container-fluid">
19             <a class="navbar-brand" href="{{ url_for('users.index') }}">
20                 User Management
21             </a>
22         </div>
23     </nav>
24
25     <main class="container">
26         {% with messages = get_flashed_messages(with_categories=true) %}
27         {% if messages %}
28             {% for category, message in messages %}
29                 <div class="alert alert-{{ category }} alert-dismissible fade show" role="alert">
30                     {{ message }}
31                     <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
32                 </div>
33             {% endfor %}
34             {% endif %}
35         {% endwith %}
36
37         {% block content %}{% endblock %}
38     </main>
39
40     <!-- Bootstrap Bundle JS -->
41     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
42     <script src="{{ url_for('static', filename='js/main.js') }}"></script>
43     {% block extra_js %}{% endblock %}
44 </body>
45 </html>
```

FLASK_RBAC_SQLITE3 .venvUsers app forms _init_.py user_forms.py models _init_.py user.py routes _init_.py user_routes.py services _init_.py user_service.py static css images js main.js templates layouts base.html permissions roles users _form.html create.html delete_confirm.html detail.html edit.html index.html _init_.py instance users.db migrations tests utils config.py extensions.py requirements.txt run.py

app > templates > users > _form.html

```

1  {{ form.hidden_tag() }}
2
3  <div class="mb-3">
4      {{ form.username.label(class="form-label") }}
5      {{ form.username(class="form-control") }}
6      {% if form.username.errors %}
7          <div class="text-danger small">
8              {% for error in form.username.errors %}
9                  <div>{{ error }}</div>
10                 {% endfor %}
11             </div>
12         {% endif %}
13     </div>
14
15     <div class="mb-3">
16         {{ form.email.label(class="form-label") }}
17         {{ form.email(class="form-control") }}
18         {% if form.email.errors %}
19             <div class="text-danger small">
20                 {% for error in form.email.errors %}
21                     <div>{{ error }}</div>
22                     {% endfor %}
23                 </div>
24             {% endif %}
25         </div>
26
27     <div class="mb-3">
28         {{ form.full_name.label(class="form-label") }}
29         {{ form.full_name(class="form-control") }}
30         {% if form.full_name.errors %}
31             <div class="text-danger small">
32                 {% for error in form.full_name.errors %}
33                     <div>{{ error }}</div>
34                     {% endfor %}
35                 </div>
36             {% endif %}
37         </div>

```

```

38
39     <div class="form-check mb-3">
40         {{ form.is_active(class="form-check-input") }}
41         {{ form.is_active.label(class="form-check-label") }}
42     </div>
43
44     # Password fields (may be required or optional depending on form class)
45     <div class="mb-3">
46         {{ form.password.label(class="form-label") }}
47         {{ form.password(class="form-control") }}
48         {% if form.password.errors %}
49             <div class="text-danger small">
50                 {% for error in form.password.errors %}
51                     <div>{{ error }}</div>
52                     {% endfor %}
53                 </div>
54             {% endif %}
55         </div>
56
57     <div class="mb-3">
58         {{ form.confirm_password.label(class="form-label") }}
59         {{ form.confirm_password(class="form-control") }}
60         {% if form.confirm_password.errors %}
61             <div class="text-danger small">
62                 {% for error in form.confirm_password.errors %}
63                     <div>{{ error }}</div>
64                     {% endfor %}
65                 </div>
66             {% endif %}
67         </div>
68
69     <div class="mt-4">
70         {{ form.submit(class="btn btn-primary") }}
71         <a href="{{ url_for('users.index') }}" class="btn btn-outline-secondary">
72             Cancel
73         </a>
74     </div>

```

FLASK_RBAC_SQLITE3

- .venvUsers
- app
 - forms
 - _init_.py
 - user_forms.py
 - models
 - _init_.py
 - user.py
 - routes
 - _init_.py
 - user_routes.py
 - services
 - _init_.py
 - user_service.py
- static
 - css
 - images
 - js
 - main.js
- templates
 - layouts
 - base.html
 - permissions
 - roles
 - users
 - _form.html
 - create.html
 - delete_confirm.html
 - detail.html
 - edit.html
 - index.html
- _init_.py
- instance
 - users.db
- migrations
- tests
- utils
 - config.py
 - extensions.py
- requirements.txt
- run.py

app > templates > users > create.html

```

1  {% extends "layouts/base.html" %}
2  {% block title %}Create User{% endblock %}
3
4  {% block content %}
5  <h1 class="h3 mb-3">Create User</h1>
6  <form method="post" novalidate>
7  |   {% include "users/_form.html" %}
8  </form>
9  {% endblock %}

```

Edit User

User Management

Username: admin

Email: admin@email.com

Full name: SEK SOCHEAT

Active

New password (leave blank to keep current)

New strong password (optional)

Confirm new password

Update **Cancel**

app > templates > users > edit.html

```

1  {% extends "layouts/base.html" %}
2  {% block title %}Edit User{% endblock %}
3
4  {% block content %}
5  <h1 class="h3 mb-3">Edit User</h1>
6  <form method="post" novalidate>
7  |   {% include "users/_form.html" %}
8  </form>
9  {% endblock %}

```

FLASK_RBAC_SQLITE3

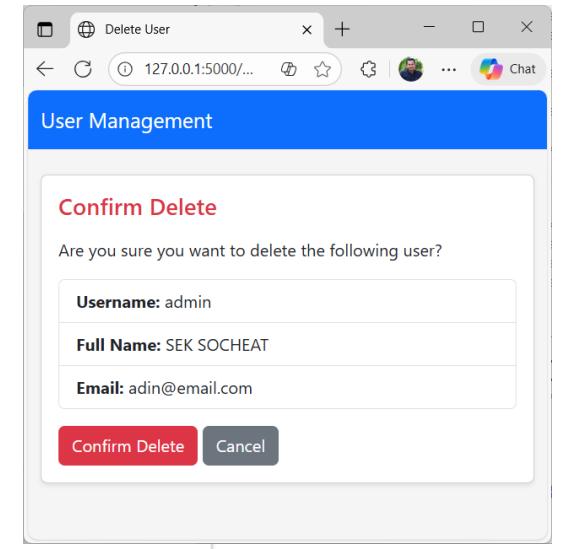
- .venvUsers
- app
 - forms
 - _init_.py
 - user_forms.py
 - models
 - _init_.py
 - user.py
 - routes
 - _init_.py
 - user_routes.py
 - services
 - _init_.py
 - user_service.py
- static
 - css
 - images
 - js
 - main.js
- templates
 - layouts
 - base.html
 - permissions
 - roles
 - users
 - _form.html
 - create.html
 - delete_confirm.html
 - detail.html
 - edit.html
 - index.html
- _init_.py
- instance
 - users.db
- migrations
- tests
- utils
- config.py
- extensions.py
- requirements.txt
- run.py

app > templates > users > delete_confirm.html

```

1   {% extends "layouts/base.html" %}
2   {% block title %}Delete User{% endblock %}
3
4   {% block content %}
5
6   <div class="card shadow-sm">
7       <div class="card-body">
8           <h3 class="h4 text-danger">Confirm Delete</h3>
9
10      <p class="mt-3">Are you sure you want to delete the following user?</p>
11
12      <ul class="list-group mb-3">
13          <li class="list-group-item"><strong>Username:</strong> {{ user.username }}</li>
14          <li class="list-group-item"><strong>Full Name:</strong> {{ user.full_name }}</li>
15          <li class="list-group-item"><strong>Email:</strong> {{ user.email }}</li>
16      </ul>
17
18      <form method="POST" action="{{ url_for('users.delete', user_id=user.id) }}>
19          {{ form.hidden_tag() }}
20          {{ form.submit(class="btn btn-danger") }}
21          <a href="{{ url_for('users.index') }}" class="btn btn-secondary">Cancel</a>
22      </form>
23  </div>
24</div>
25
26  {% endblock %}

```



FLASK_RBAC_SQLITE3 .venvUsers app forms _init_.py user_forms.py models _init_.py user.py routes _init_.py user_routes.py services _init_.py user_service.py static css images js main.js templates layouts base.html permissions roles users _form.html create.html delete_confirm.html detail.html edit.html index.html _init_.py instance users.db migrations tests utils config.py extensions.py requirements.txt run.py

app > templates > users > detail.html

```

1  {% extends "layouts/base.html" %}
2  {% block title %}User {{ user.username }}{% endblock %}
3
4  {% block content %}
5  <div class="d-flex justify-content-between align-items-center mb-3">
6    <h1 class="h3 mb-0">User Detail</h1>
7    <div>
8      <a href="{{ url_for('users.edit', user_id=user.id) }}" class="btn btn-sm btn-outline-secondary">Edit</a>
9      <a href="{{ url_for('users.index') }}" class="btn btn-sm btn-outline-primary">Back to list</a>
10     </div>
11   </div>
12
13  <dl class="row">
14    <dt class="col-sm-3">Username</dt>
15    <dd class="col-sm-9">{{ user.username }}</dd>
16
17    <dt class="col-sm-3">Full name</dt>
18    <dd class="col-sm-9">{{ user.full_name }}</dd>
19
20    <dt class="col-sm-3">Email</dt>
21    <dd class="col-sm-9">{{ user.email }}</dd>
22
23    <dt class="col-sm-3">Status</dt>
24    <dd class="col-sm-9">
25      {% if user.is_active %}
26        <span class="badge text-bg-success">Active</span>
27      {% else %}
28        <span class="badge text-bg-secondary">Inactive</span>
29      {% endif %}
30    </dd>
31
32    <dt class="col-sm-3">Created at</dt>
33    <dd class="col-sm-9">{{ user.created_at }}</dd>
34
35    <dt class="col-sm-3">Last updated</dt>
36    <dd class="col-sm-9">{{ user.updated_at }}</dd>
37  </dl>
38  {% endblock %}

```

User Management

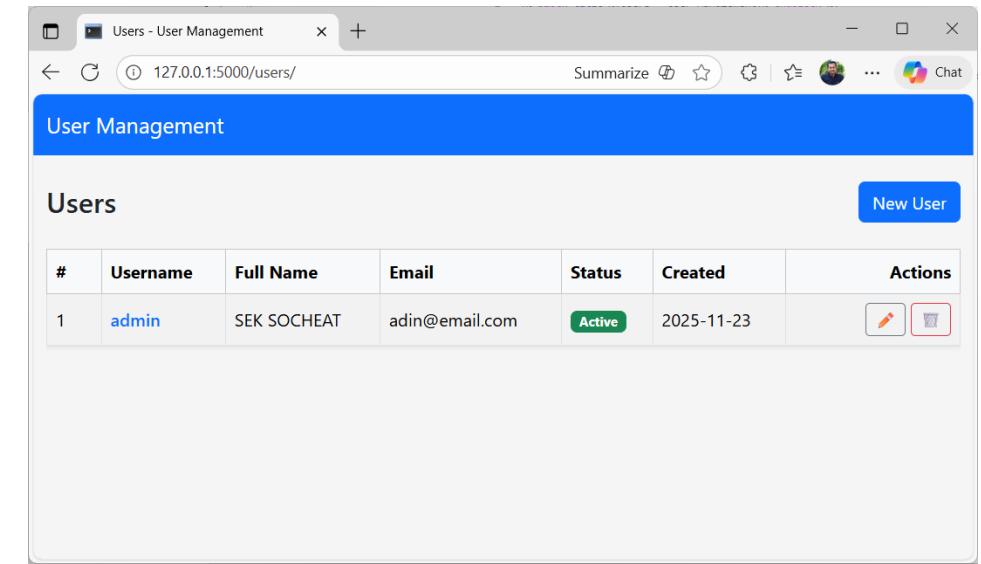
User 'admin' was updated successfully.

User Detail

Username	admin
Full name	SEK SOCHEAT
Email	admin@email.com
Status	Active
Created at	2025-11-23 16:06:35.527711
Last updated	2025-11-25 04:55:41.835470

- ✓ FLASK_RBAC_SQLITE3
 - > .venvUsers
 - ✓ app
 - ✓ forms
 - ➊ __init__.py
 - ➋ user_forms.py
 - ✓ models
 - ➊ __init__.py
 - ➋ user.py
 - ✓ routes
 - ➊ __init__.py
 - ➋ user_routes.py
 - ✓ services
 - ➊ __init__.py
 - ➋ user_service.py
 - ✓ static
 - > css
 - > images
 - ✓ js
 - JS main.js
 - ✓ templates
 - ✓ layouts
 - ➊ base.html
 - ✓ permissions
 - ✓ roles
 - ✓ users
 - ➊ _form.html
 - ➋ create.html
 - ➋ delete_confirm.html
 - ➋ detail.html
 - ➋ edit.html
 - ➋ index.html
 - ➊ __init__.py
 - ✓ instance
 - ☰ users.db
 - ✓ migrations
 - > tests
 - > utils
 - ➊ config.py
 - ➊ extensions.py
 - ☰ requirements.txt
 - ➊ run.py

```
app > templates > users > index.html
1  {% extends "layouts/base.html" %} 
2  {% block title %}Users - User Management{% endblock %} 
3
4  {% block content %} 
5  <div class="d-flex justify-content-between align-items-center mb-4">
6    <h1 class="h3 mb-0">Users</h1>
7    <a href="{{ url_for('users.create') }}" class="btn btn-primary">New User</a>
8  </div>
9
10 {% if users %} 
11   <div class="table-responsive">
12     <table class="table table-striped table-bordered align-middle shadow-sm">
13       <thead class="table-light">
14         <tr>
15           <th style="width: 50px;">#</th>
16           <th>Username</th>
17           <th>Full Name</th>
18           <th>Email</th>
19           <th>Status</th>
20           <th>Created</th>
21           <th class="text-end" style="width: 160px;">Actions</th>
22         </tr>
23       </thead>
24       <tbody>
25         {% for user in users %} 
26           <tr>
27             <td>{{ loop.index }}</td>
28
29             <td>
30               <a
31                 href="{{ url_for('users.detail', user_id=user.id) }}"
32                 class="text-decoration-none fw-semibold"
33               >
34                 {{ user.username }}
35               </a>
36             </td>
37
38             <td>{{ user.full_name }}</td>
39             <td>{{ user.email }}</td>
40           </tr>
41         {% endfor %}
42       </tbody>
43     </table>
44   </div>
45
46   <div class="mt-4">
47     <p>Total: {{ users|length }} users</p>
48     <ul class="list-group list-group-flush">
49       {% for user in users %} 
50         <li>{{ user.username }} ({{ user.full_name }})</li>
51       {% endfor %}
52     </ul>
53   </div>
54
55 </div>
```



FLASK_RBAC_SQLITE3

- .venvUsers
- app
 - forms
 - _init_.py
 - user_forms.py
 - models
 - _init_.py
 - user.py
 - routes
 - _init_.py
 - user_routes.py
 - services
 - _init_.py
 - user_service.py
 - static
 - css
 - images
 - js
 - main.js
- templates
 - layouts
 - base.html
 - permissions
 - roles
 - users
 - _form.html
 - create.html
 - delete_confirm.html
 - detail.html
 - edit.html
 - index.html
- _init_.py
- instance
 - users.db
- migrations
- tests
- utils
- config.py
- extensions.py
- requirements.txt
- run.py

```

40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82

```

```

<td>
  {% if user.is_active %}
    <span class="badge bg-success">Active</span>
  {% else %}
    <span class="badge bg-secondary">Inactive</span>
  {% endif %}
</td>

<td>{{ user.created_at.strftime('%Y-%m-%d') }}</td>

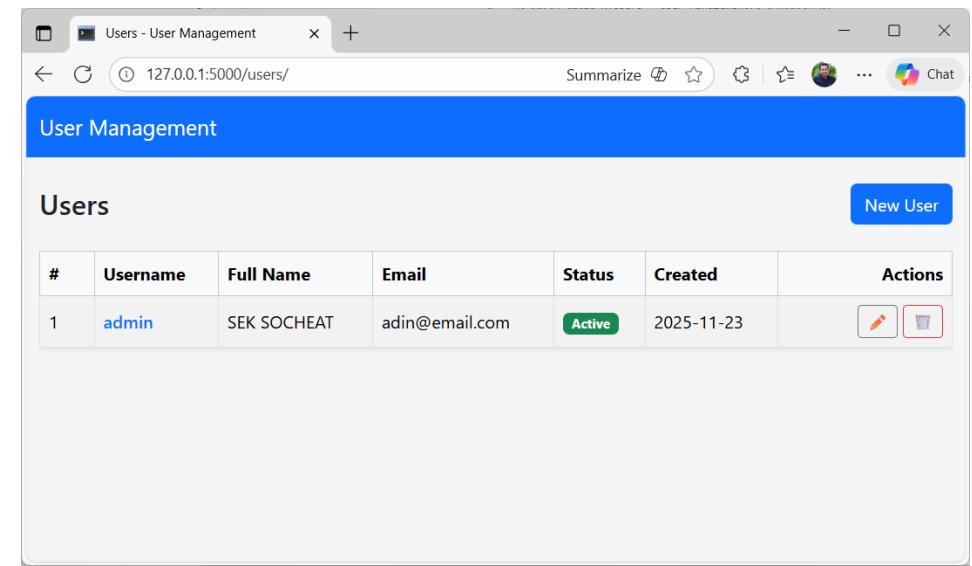
<td class="text-end">
  <!-- Edit -->
  <a href="{{ url_for('users.edit', user_id=user.id) }}"
    class="btn btn-sm btn-outline-secondary"
    title="Edit">
    <img alt="Edit icon" />
  </a>

  <!-- Delete (go to custom delete confirmation page) -->
  <a href="{{ url_for('users.delete_confirm', user_id=user.id) }}"
    class="btn btn-sm btn-outline-danger"
    title="Delete">
    <img alt="Delete icon" />
  </a>
</td>

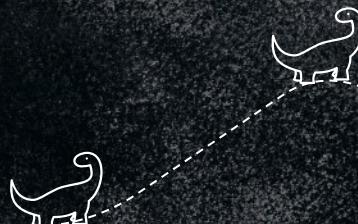
</tr>
{% endfor %}
</tbody>
</table>
</div>

{% else %}
  <div class="alert alert-info">
    No users found.
    <a href="{{ url_for('users.create') }}">Create the first user</a>.
  </div>
{% endif %}
{% endblock %}

```



3 — Users Registration in forms/



The image shows a file structure on the left and two code snippets on the right. A red arrow points from the file structure to the first code snippet, and another red arrow points from the first code snippet to the second.

File Structure:

- FLASK_RBAC_SQLITE3
- .venvUsers
- app
 - forms
 - __init__.py
 - user_forms.py
 - models
 - __init__.py
 - user.py
 - routes
 - __init__.py
 - user_routes.py
 - services
 - __init__.py
 - user_service.py
 - static
 - css
 - images
 - js
 - main.js
 - templates
 - layouts
 - base.html
 - permissions
 - roles
 - users
 - _form.html
 - create.html
 - delete_confirm.html
 - detail.html
 - edit.html
 - index.html
- instance
 - users.db
- migrations
- tests
- utils
- config.py
- extensions.py
- requirements.txt
- run.py

```
app > forms > __init__.py
1   from .user_forms import UserCreateForm, UserEditForm
2   from .user_forms import ConfirmDeleteForm
```

```
app > forms > user_forms.py > ...
1   import re
2   from flask_wtf import FlaskForm
3   from wtforms import BooleanField, StringField, SubmitField, PasswordField
4   from wtforms.validators import DataRequired, Email, Length, EqualTo, ValidationError
5
6   from app.models import User
7   from extensions import db
8
9   # ----- helpers -----
10
11 def strong_password(form, field):
12     """Require: min 8 chars, upper, lower, digit, special."""
13     password = field.data or ""
14     if len(password) < 8:
15         raise ValidationError("Password must be at least 8 characters long.")
16
17     if not re.search(r"[A-Z]", password):
18         raise ValidationError("Password must contain at least one uppercase letter.")
19
20     if not re.search(r"[a-z]", password):
21         raise ValidationError("Password must contain at least one lowercase letter.")
22
23     if not re.search(r"[0-9]", password):
24         raise ValidationError("Password must contain at least one digit.")
25
26     if not re.search(r"[@#$%^&*(),.?\":{}|>_\\-+]", password):
27         raise ValidationError("Password must contain at least one special character.")
```

FLASK_RBAC_SQLITE3
|.venvUsers
app
forms
 __init__.py
 user_forms.py
models
 __init__.py
 user.py
routes
 __init__.py
 user_routes.py
services
 __init__.py
 user_service.py
static
 css
 images
 js
 main.js
templates
 layouts
 base.html
 permissions
 roles
 users
 _form.html
 create.html
 delete_confirm.html
 detail.html
 edit.html
 index.html
 __init__.py
instance
 users.db
migrations
tests
utils
 config.py
 extensions.py
requirements.txt
run.py

```
30 # ----- create form (password required) -----
31
32 class UserCreateForm(FlaskForm):
33     username = StringField(
34         "Username",
35         validators=[DataRequired(), Length(min=3, max=80)],
36         render_kw={"placeholder": "Enter username"},
37     )
38     email = StringField(
39         "Email",
40         validators=[DataRequired(), Email(), Length(max=120)],
41         render_kw={"placeholder": "Enter email"},
42     )
43     full_name = StringField(
44         "Full name",
45         validators=[DataRequired(), Length(min=3, max=120)],
46         render_kw={"placeholder": "Enter full name"},
47     )
48     is_active = BooleanField("Active", default=True)
49
50     password = PasswordField(
51         "Password",
52         validators=[
53             DataRequired(),
54             strong_password,
55         ],
56         render_kw={"placeholder": "Strong password"},
57     )
58     confirm_password = PasswordField(
59         "Confirm password",
60         validators=[
61             DataRequired(),
62             EqualTo("password", message="Passwords must match."),
63         ],
64         render_kw={"placeholder": "Confirm password"},
65     )
66
67     submit = SubmitField("Save")
68
```

FLASK_RBAC_SQLITE3 .venvUsers
app forms
 __init__.py user_forms.py
models
 __init__.py user.py
routes
 __init__.py user_routes.py
services
 __init__.py user_service.py
static
 css
 images
 js
 main.js
templates
 layouts
 base.html
 permissions
 roles
 users
 _form.html
 create.html
 delete_confirm.html
 detail.html
 edit.html
 index.html
 __init__.py
instance users.db
migrations
tests
utils config.py
extensions.py
requirements.txt
run.py

```
69 # ----- server-side uniqueness checks -----
70
71 def validate_username(self, field):
72     exists = db.session.scalar(
73         db.select(User).filter(User.username == field.data)
74     )
75     if exists:
76         raise ValidationError("This username is already taken.")
77
78 def validate_email(self, field):
79     exists = db.session.scalar(
80         db.select(User).filter(User.email == field.data)
81     )
82     if exists:
83         raise ValidationError("This email is already registered.")
84
85
```



FLASK_RBAC_SQLITE3 ...

.venvUsers

app

forms

 __init__.py

 user_forms.py

models

 __init__.py

 user.py

routes

 __init__.py

 user_routes.py

services

 __init__.py

 user_service.py

static

 css

 images

 js

 main.js

templates

 layouts

 base.html

 permissions

 roles

 users

 _form.html

 create.html

 delete_confirm.html

 detail.html

 edit.html

 index.html

 __init__.py

instance

 users.db

migrations

tests

utils

config.py

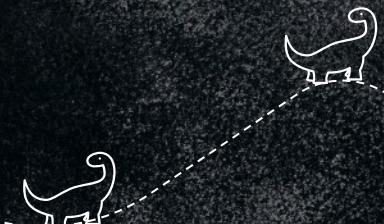
extensions.py

requirements.txt

run.py

```
86 # ----- edit form (password optional) -----
87
88 class UserEditForm(FlaskForm):
89     username = StringField(
90         "Username",
91         validators=[DataRequired(), Length(min=3, max=80)],
92     )
93     email = StringField(
94         "Email",
95         validators=[DataRequired(), Email(), Length(max=120)],
96     )
97     full_name = StringField(
98         "Full name",
99         validators=[DataRequired(), Length(min=3, max=120)],
100    )
101    is_active = BooleanField("Active")
102
103    # optional password - only change if filled
104    password = PasswordField(
105        "New password (leave blank to keep current)",
106        validators=[strong_password],
107        render_kw={"placeholder": "New strong password (optional)"}
108    )
109    confirm_password = PasswordField(
110        "Confirm new password",
111        validators=[EqualTo("password", message="Passwords must match.")],
112    )
113
114    submit = SubmitField("Update")
115
116    def __init__(self, original_user: User, *args, **kwargs):
117        super().__init__(*args, **kwargs)
118        self.original_user = original_user
119
120    def validate_username(self, field):
121        q = db.select(User).filter(User.username == field.data, User.id != self.original_user.id)
122        exists = db.session.scalar(q)
123        if exists:
124            raise ValidationError("This username is already taken.")
125
126    def validate_email(self, field):
127        q = db.select(User).filter(User.email == field.data, User.id != self.original_user.id)
128        exists = db.session.scalar(q)
129        if exists:
130            raise ValidationError("This email is already registered.")
131
132    class ConfirmDeleteForm(FlaskForm):
133        submit = SubmitField("Confirm Delete")
```

4 — Users Registration in services /

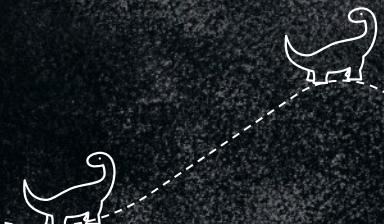


```
app > services > + _init_.py  
1   from .user_service import UserService
```

```
app > services > user_service.py > ...
1  from typing import List, Optional
2  from app.models.user import User
3  from extensions import db
4
5  class UserService:
6      @staticmethod
7      def get_all() -> List[User]:
8          return User.query.order_by(User.id.desc()).all()
9
10     @staticmethod
11     def get_by_id(user_id: int) -> Optional[User]:
12         return User.query.get(user_id)
13
14     @staticmethod
15     def create(data: dict, password: str) -> User:
16         user = User(
17             username=data["username"],
18             email=data["email"],
19             full_name=data["full_name"],
20             is_active=data.get("is_active", True),
21         )
22         user.set_password(password)
23         db.session.add(user)
24         db.session.commit()
25         return user
26
27     @staticmethod
28     def update(user: User, data: dict, password: Optional[str] = None) -> User:
29         user.username = data["username"]
30         user.email = data["email"]
31         user.full_name = data["full_name"]
32         user.is_active = data.get("is_active", True)
33
34         if password:
35             user.set_password(password)
36
37         db.session.commit()
38         return user
39
40     @staticmethod
41     def delete(user: User) -> None:
42         db.session.delete(user)
43         db.session.commit()
```



5 — Users Registration in routes/



FLASK_RBAC_SQLITE3
|.venvUsers
app
| forms
| | _init_.py
| | user_forms.py
| models
| | _init_.py
| | user.py
| routes
| | _init_.py
| | user_routes.py
| services
| | _init_.py
| | user_service.py
static
| css
| images
| js
| | main.js
templates
| layouts
| | base.html
| permissions
| roles
| users
| | _form.html
| | create.html
| | delete_confirm.html
| | detail.html
| | edit.html
| | index.html
| | _init_.py
instance
| users.db
migrations
tests
utils
config.py
extensions.py
requirements.txt
run.py

app > routes > **_init_.py**
1 from .user_routes import user_bp

app > routes > **user_routes.py** edit
1 from flask import (
2 Blueprint,
3 render_template,
4 redirect,
5 url_for,
6 flash,
7 abort,
8)
9
10 from app.forms.user_forms import UserCreateForm, UserEditForm, ConfirmDeleteForm
11 from app.services.user_service import UserService
12
13 user_bp = Blueprint("users", __name__, url_prefix="/users")
14
15 @user_bp.route("/")
16 def index():
17 users = UserService.get_all()
18 return render_template("users/index.html", users=users)
19
20 @user_bp.route("/<int:user_id>")
21 def detail(user_id: int):
22 user = UserService.get_by_id(user_id)
23 if user is None:
24 abort(404)
25 return render_template("users/detail.html", user=user)
26
27 @user_bp.route("/create", methods=["GET", "POST"])
28 def create():
29 form = UserCreateForm()
30 if form.validate_on_submit():
31 data = {
32 "username": form.username.data,
33 "email": form.email.data,
34 "full_name": form.full_name.data,
35 "is_active": form.is_active.data,
36 }
37 password = form.password.data
38 user = UserService.create(data, password)
39 flash(f"User '{user.username}' was created successfully.", "success")
40 return redirect(url_for("users.index"))
41
42 return render_template("users/create.html", form=form)

FLASK_RBAC_SQLITE3 ...

.venvUsers

app

forms

init.py

user_forms.py

models

init.py

user.py

routes

init.py

user_routes.py

services

init.py

user_service.py

static

css

images

js

main.js

templates

layouts

base.html

permissions

roles

users

_form.html

create.html

delete_confirm.html

detail.html

edit.html

index.html

init.py

instance

users.db

migrations

tests

utils

config.py

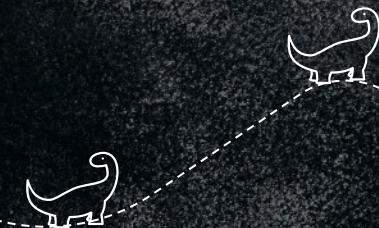
extensions.py

requirements.txt

run.py

```
43
44 @user_bp.route("/<int:user_id>/edit", methods=["GET", "POST"])
45 def edit(user_id: int):
46     user = UserService.get_by_id(user_id)
47     if user is None:
48         abort(404)
49
50     form = UserEditForm(original_user=user, obj=user)
51
52     if form.validate_on_submit():
53         data = {
54             "username": form.username.data,
55             "email": form.email.data,
56             "full_name": form.full_name.data,
57             "is_active": form.is_active.data,
58         }
59         password = form.password.data or None
60         UserService.update(user, data, password)
61         flash(f"User '{user.username}' was updated successfully.", "success")
62         return redirect(url_for("users.detail", user_id=user.id))
63
64     return render_template("users/edit.html", form=form, user=user)
65
66 @user_bp.route("/<int:user_id>/delete", methods=["GET"])
67 def delete_confirm(user_id: int):
68     user = UserService.get_by_id(user_id)
69     if user is None:
70         abort(404)
71     form = ConfirmDeleteForm()
72     return render_template("users/delete_confirm.html", user=user, form=form)
73
74
75 @user_bp.route("/<int:user_id>/delete", methods=["POST"])
76 def delete(user_id: int):
77     user = UserService.get_by_id(user_id)
78     if user is None:
79         abort(404)
80
81     UserService.delete(user)
82     flash("User was deleted successfully.", "success")
83     return redirect(url_for("users.index"))
```

6 — Users Registration in **static/**



FLASK_RBAC_SQLITE3
 .venvUsers
 app
 forms
 __init__.py
 user_forms.py
 models
 __init__.py
 user.py
 routes
 __init__.py
 user_routes.py
 services
 __init__.py
 user_service.py
 static
 > css
 > images
 js
 JS main.js
 templates
 layouts
 base.html
 permissions
 roles
 users
 _form.html
 create.html
 delete_confirm.html
 detail.html
 edit.html
 index.html
 __init__.py
 instance
 users.db
 migrations
 tests
 utils
 config.py
 extensions.py
 requirements.txt
 run.py

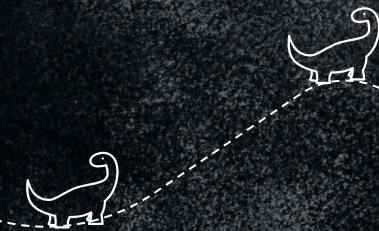
style.css

```
app > static > css > # style.css > ...
1 body {
2   background-color: #f5f5f5;
3 }
4
5 main.container {
6   max-width: 960px;
7 }
8
9 .table td,
10 .table th {
11   vertical-align: middle;
12 }
```

app > static > js > JS main.js > ...

```
1 document.addEventListener("DOMContentLoaded", () => {
2   const pwd = document.querySelector('input[name="password"]');
3   if (!pwd) return;
4
5   pwd.addEventListener("input", () => {
6     const msg = document.querySelector("#passwordHelp");
7     if (!msg) return;
8
9     const v = pwd.value;
10    const strong =
11      v.length >= 8 &&
12      /[A-Z]/.test(v) &&
13      /[a-z]/.test(v) &&
14      /[0-9]/.test(v) &&
15      /[!@#$%^&*(),.?":{}|<>\-+=]/.test(v);
16
17    msg.textContent = strong
18      ? "Strong password ✅"
19      : "Use at least 8 chars, with upper, lower, number, and special symbol.";
20    msg.className = strong ? "form-text text-success" : "form-text text-danger";
21  });
22});
```

7 — app/__init__.py , run.py , config.py , ...



FLASK_RBAC_SQLITE3

- .venvUsers
- app
 - forms
 - _init_.py
 - user_forms.py
 - models
 - _init_.py
 - user.py
 - routes
 - _init_.py
 - user_routes.py
 - services
 - _init_.py
 - user_service.py
 - static
 - css
 - images
 - js
 - main.js
 - templates
 - layouts
 - base.html
 - permissions
 - roles
 - users
 - _form.html
 - create.html
 - delete_confirm.html
 - detail.html
 - edit.html
 - index.html
 - _init_.py
- instance
 - users.db
- migrations
- tests
- utils
- config.py
- extensions.py
- requirements.txt
- run.py

```
app > _init_.py > ...
1   from flask import Flask, redirect, url_for
2   from config import Config
3   from extensions import db, csrf
4
5   def create_app(config_class: type[Config] = Config):
6       app = Flask(__name__)
7       app.config.from_object(config_class)
8
9       db.init_app(app)
10      csrf.init_app(app)
11
12      # Register blueprints
13      from app.routes.user_routes import user_bp
14      app.register_blueprint(user_bp)
15
16      # 👉 Add this block so "/" goes to the Users list
17      @app.route("/")
18      def home():
19          return redirect(url_for("users.index"))
20
21      # Create tables
22      with app.app_context():
23          from app.models import User # noqa: F401
24          db.create_all()
25
26      return app
```

```
run.py > ...
1   from app import create_app
2
3   app = create_app()
4
5   if __name__ == "__main__":
6       app.run(debug=True)
7
```



Thank You

SEK SocheaT

✉ socheatsek@norton-u.com

