Создание админки с авторизацией и возможностью выставлять счета требует разработки бэкенда и фронтенда. Ниже я предоставлю простую структуру для веб-приложения, используя Python с Flask для бэкенда и HTML с некоторым JavaScript для фронтенда. Также будет использована SQLite для хранения данных.

### 1. Установка необходимых библиотек

Сначала установите необходимые библиотеки:

```bash

pip install Flask Flask-SQLAlchemy Flask-Login

```

### 2. Структура проекта

Создайте следующую структуру папок:

```

admin\_panel/

├── app.py

├── models.py

├── templates/

│ ├── login.html

│ ├── admin.html

│ ├── invoice.html

└── static/

└── styles.css

```

### 3. Модели (models.py)

Создайте файл `models.py` для определения модели пользователя и счета:

```python

from flask\_sqlalchemy import SQLAlchemy

from flask\_login import UserMixin

db = SQLAlchemy()

class User(db.Model, UserMixin):

id = db.Column(db.Integer, primary\_key=True)

username = db.Column(db.String(150), unique=True, nullable=False)

password = db.Column(db.String(150), nullable=False) # Лучше использовать хеширование

class Invoice(db.Model):

id = db.Column(db.Integer, primary\_key=True)

amount = db.Column(db.Float, nullable=False)

details = db.Column(db.String(500), nullable=False)

user\_id = db.Column(db.Integer, db.ForeignKey('user.id'))

```

### 4. Основной файл приложения (app.py)

Создайте файл `app.py`:

```python

from flask import Flask, render\_template, redirect, url\_for, request, flash

from flask\_sqlalchemy import SQLAlchemy

from flask\_login import LoginManager, UserMixin, login\_user, logout\_user, login\_required, current\_user

from models import db, User, Invoice

app = Flask(\_\_name\_\_)

app.config['SECRET\_KEY'] = 'your\_secret\_key'

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///site.db'

db.init\_app(app)

login\_manager = LoginManager(app)

login\_manager.login\_view = 'login'

@login\_manager.user\_loader

def load\_user(user\_id):

return User.query.get(int(user\_id))

with app.app\_context():

db.create\_all()

@app.route('/login', methods=['GET', 'POST'])

def login():

if request.method == 'POST':

username = request.form['username']

password = request.form['password']

user = User.query.filter\_by(username=username).first()

if user and user.password == password: # Здесь следует использовать хеширование

login\_user(user)

return redirect(url\_for('admin'))

flash('Invalid username or password')

return render\_template('login.html')

@app.route('/logout')

@login\_required

def logout():

logout\_user()

return redirect(url\_for('login'))

@app.route('/admin')

@login\_required

def admin():

invoices = Invoice.query.all()

return render\_template('admin.html', invoices=invoices)

@app.route('/add\_invoice', methods=['POST'])

@login\_required

def add\_invoice():

amount = request.form['amount']

details = request.form['details']

new\_invoice = Invoice(amount=amount, details=details, user\_id=current\_user.id)

db.session.add(new\_invoice)

db.session.commit()

return redirect(url\_for('admin'))

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

```

### 5. Шаблоны (templates/login.html, templates/admin.html)

\*\*login.html\*\*

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login</title>

</head>

<body>

<form method="POST">

<h2>Login</h2>

<input type="text" name="username" placeholder="Username" required>

<input type="password" name="password" placeholder="Password" required>

<button type="submit">Login</button>

</form>

</body>

</html>

```

\*\*admin.html\*\*

```html

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Admin Panel</title>

</head>

<body>

<h2>Admin Panel</h2>

<a href="{{ url\_for('logout') }}">Logout</a>

<h3>Invoices</h3>

<form method="POST" action="{{ url\_for('add\_invoice') }}">

<input type="text" name="amount" placeholder="Amount" required>

<input type="text" name="details" placeholder="Details" required>

<button type="submit">Add Invoice</button>

</form>

<ul>

{% for invoice in invoices %}

<li>Invoice {{ invoice.id }}: {{ invoice.amount }} - {{ invoice.details }}</li>

{% endfor %}

</ul>

</body>

</html>

```

### 6. Запуск

Теперь запустите ваше приложение:

```bash

python app.py

```

Теперь вы сможете зайти на свой сайт по адресу `http://127.0.0.1:5000/login`, авторизоваться и добавлять счета в админке.

### Примечание

- Для фактической разработки необходимо реализовать хеширование паролей при помощи библиотеки, такой как `werkzeug.security`.

- Следует добавить обработку ошибок и более сложные функции для управления пользователями и счетами.

- Для продакшен-версии стоит использовать более надежную базу данных (например, PostgreSQL) и настроить сервер для хостинга.

- Это базовая структура, на основе которой можно создавать более сложные системы.