Adoption Shelter Solution

Download Solution

part-1/models.py

```
"""Models for adopt app."""

from flask_sqlalchemy import SQLAlchemy

GENERIC_IMAGE = "https://mylostpetalert.com/wp-content/themes/mlpa-child/images/nophoto.gif"

db = SQLAlchemy()


class Pet(db.Model):
    """Adoptable pet."""

    __tablename__ = "pets"

    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.Text, nullable=False)
    species = db.Column(db.Text, nullable=False)
    photo_url = db.Column(db.Text)
    age = db.Column(db.Integer)
    notes = db.Column(db.Text)
    available = db.Column(db.Boolean, nullable=False, default=True)

    def image_url(self):
        """Return image for pet -- bespoke or generic."""

        return self.photo_url or GENERIC_IMAGE


def connect_db(app):
    """Connect this database to provided Flask app.

    You should call this in your Flask app.
    """

    db.app = app
    db.init_app(app)
```

part-1/app.py

```
"""Flask app for adopt app."""

from flask import Flask, url_for, render_template, redirect, flash, jsonify

from flask_debugtoolbar import DebugToolbarExtension

from models import db, connect_db, Pet
from forms import AddPetForm, EditPetForm

app = Flask(__name__)

app.config['SECRET_KEY'] = "abcdef"

app.config['SQLALCHEMY_DATABASE_URI'] = "postgresql:///adopt"
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

connect_db(app)
db.create_all()

# Having the Debug Toolbar show redirects explicitly is often useful;
# however, if you want to turn it off, you can uncomment this line:
#
# app.config['DEBUG_TB_INTERCEPT_REDIRECTS'] = False

toolbar = DebugToolbarExtension(app)


######################################################################


@app.route("/")
def list_pets():
    """List all pets."""

    pets = Pet.query.all()
    return render_template("pet_list.html", pets=pets)


@app.route("/add", methods=["GET", "POST"])
def add_pet():
    """Add a pet."""

    form = AddPetForm()

    if form.validate_on_submit():
        data = {k: v for k, v in form.data.items() if k != "csrf_token"}
        new_pet = Pet(**data)
        # new_pet = Pet(name=form.name.data, age=form.age.data, ...)
        db.session.add(new_pet)
        db.session.commit()
        flash(f"{new_pet.name} added.")
        return redirect(url_for('list_pets'))

    else:
        # re-present form for editing
        return render_template("pet_add_form.html", form=form)


@app.route("/<int:pet_id>", methods=["GET", "POST"])
def edit_pet(pet_id):
    """Edit pet."""

    pet = Pet.query.get_or_404(pet_id)
    form = EditPetForm(obj=pet)

    if form.validate_on_submit():
        pet.notes = form.notes.data
        pet.available = form.available.data
        pet.photo_url = form.photo_url.data
        db.session.commit()
        flash(f"{pet.name} updated.")
        return redirect(url_for('list_pets'))

    else:
        # failed; re-present form for editing
        return render_template("pet_edit_form.html", form=form, pet=pet)


@app.route("/api/pets/<int:pet_id>", methods=['GET'])
def api_get_pet(pet_id):
    """Return basic info about pet in JSON."""

    pet = Pet.query.get_or_404(pet_id)
    info = {"name": pet.name, "age": pet.age}

    return jsonify(info)
```

part-1/forms.py

```
"""Forms for adopt app."""

from flask_wtf import FlaskForm
from wtforms import StringField, IntegerField, SelectField, TextAreaField, BooleanField
from wtforms.validators import InputRequired, Length, NumberRange, URL, Optional


class AddPetForm(FlaskForm):
    """Form for adding pets."""

    name = StringField(
        "Pet Name",
        validators=[InputRequired()],
    )

    species = SelectField(
        "Species",
        choices=[("cat", "Cat"), ("dog", "Dog"), ("porcupine", "Porcupine")],
    )

    photo_url = StringField(
        "Photo URL",
        validators=[Optional(), URL()],
    )

    age = IntegerField(
        "Age",
        validators=[Optional(), NumberRange(min=0, max=30)],
    )

    notes = TextAreaField(
        "Comments",
        validators=[Optional(), Length(min=10)],
    )


class EditPetForm(FlaskForm):
    """Form for editing an existing pet."""

    photo_url = StringField(
        "Photo URL",
        validators=[Optional(), URL()],
    )

    notes = TextAreaField(
        "Comments",
        validators=[Optional(), Length(min=10)],
    )

    available = BooleanField("Available?")
```

HTML

part-1/templates/_form.html

```
{{ form.hidden_tag() }}

{% for field in form if field.widget.input_type != 'hidden'%}

<div class="form-group">
  {{ field.label }}
  {{ field(class="form-control") }}

  <span>
      {% if field.errors %}
        {% for error in field.errors %}
                <b class="text-danger">{{ error }}</b>
        {% endfor %}
      {% endif %}
    </span>

</div>

{% endfor %}
```

part-1/templates/pet_list.html

```
{% extends 'base.html' %}

{% block content %}

<h1>Our Pets</h1>

<div class="row pets-listing">

  {% for pet in pets %}
  <div class="col-3">
    <img class="img-thumbnail" src="{{ pet.image_url() }}">
    <a href="{{ url_for('edit_pet', pet_id=pet.id) }}">{{ pet.name }}</a>
    {% if pet.available %} <b> is available!</b> {% endif %}
  </div>
  {% else %}
  <div class="col-12">No pets yet.</div>
  {% endfor %}

</div>

<a class="btn btn-primary" href="{{ url_for('add_pet') }}">Add a Pet</a>

{% endblock %}
```

part-1/templates/pet_add_form.html

```
{% extends 'base.html' %}

{% block content %}

<h1>Add a Pet</h1>

<form method="POST">
  {% include "_form.html" %}
  <button class="btn btn-primary" type="submit">Add</button>
</form>

{% endblock %}
```

part-1/templates/pet_edit_form.html

```
{% extends 'base.html' %}

{% block content %}

<h1>{{ pet.name }}</h1>

{% if pet.photo_url %}
<img class="img-fluid" src="{{ pet.photo_url }}">
{% endif %}

<h2>Update</h2>

<form method="POST">
  {% include "_form.html" %}
  <button class="btn btn-primary" type="submit">Update</button>
</form>

{% endblock %}
```