--- config.py ---

import os

class Config: SECRET\_KEY = os.environ.get("SECRET\_KEY", "dev-secret-key") SQLALCHEMY\_DATABASE\_URI = os.environ.get("DATABASE\_URL", "sqlite:///soccer.db") SQLALCHEMY\_TRACK\_MODIFICATIONS = False

class ProdConfig(Config): DEBUG = False

class DevConfig(Config): DEBUG = True

--- requirements.txt ---

Flask==2.2.5 Flask-WTF==1.1.3 Flask-Login==0.6.3 Flask-SQLAlchemy==3.1.1 Werkzeug==3.0.2 email-validator==1.3.1 python-dotenv==1.0.1

--- models.py ---

from flask\_sqlalchemy import SQLAlchemy from flask\_login import UserMixin from datetime import datetime from sqlalchemy import CheckConstraint, UniqueConstraint, Enum as SAEnum, Index import enum

db = SQLAlchemy()

class RoleEnum(enum.Enum): CTO = "cto" ADMIN = "admin" SCOUT = "scout"

class GenderEnum(enum.Enum): MALE = "Male" FEMALE = "Female" OTHER = "Other"

class FootEnum(enum.Enum): LEFT = "Left" RIGHT = "Right" BOTH = "Both"

class Club(db.Model): id = db.Column(db.Integer, primary\_key=True) name = db.Column(db.String(120), unique=True, nullable=False) founded = db.Column(db.Date) nationality = db.Column(db.String(60)) badge\_url = db.Column(db.String(250)) email = db.Column(db.String(120)) phone = db.Column(db.String(40)) hide\_contact = db.Column(db.Boolean, default=True) players = db.relationship('PlayerProfile', back\_populates='club') history = db.relationship('PlayerClubHistory', back\_populates='club', cascade="all, delete-orphan")

class User(UserMixin, db.Model): id = db.Column(db.Integer, primary\_key=True) email = db.Column(db.String(120), unique=True, nullable=False) password = db.Column(db.String(200), nullable=False) role = db.Column(SAEnum(RoleEnum), default=RoleEnum.SCOUT, nullable=False) created\_at = db.Column(db.DateTime, default=datetime.utcnow) scout = db.relationship('ScoutProfile', uselist=False, back\_populates='user')

class ScoutProfile(db.Model): id = db.Column(db.Integer, primary\_key=True) user\_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False) full\_name = db.Column(db.String(120), nullable=False) nationality = db.Column(db.String(60)) photo\_url = db.Column(db.String(250)) phone = db.Column(db.String(40)) hide\_contact = db.Column(db.Boolean, default=True) user = db.relationship('User', back\_populates='scout') reports = db.relationship('ScoutingReport', back\_populates='scout', cascade="all, delete-orphan")

class PlayerProfile(db.Model): id = db.Column(db.Integer, primary\_key=True) first\_name = db.Column(db.String(60), nullable=False) middle\_name = db.Column(db.String(60)) last\_name = db.Column(db.String(60), nullable=False) dob = db.Column(db.Date, nullable=False) nationality = db.Column(db.String(60)) gender = db.Column(SAEnum(GenderEnum)) dominant\_foot = db.Column(SAEnum(FootEnum)) weak\_foot\_pct = db.Column(db.Integer) height\_cm = db.Column(db.Integer) weight\_kg = db.Column(db.Integer) photo\_url = db.Column(db.String(250)) club\_id = db.Column(db.Integer, db.ForeignKey('club.id')) club = db.relationship('Club', back\_populates='players') stats = db.relationship('PlayerMatchStat', back\_populates='player', cascade="all, delete-orphan") history = db.relationship('PlayerClubHistory', back\_populates='player', cascade="all, delete-orphan") reports = db.relationship('ScoutingReport', back\_populates='player', cascade="all, delete-orphan")

\_\_table\_args\_\_ = (

CheckConstraint('weak\_foot\_pct BETWEEN 0 AND 100', name='ck\_weak\_foot\_pct'),

Index('ix\_player\_last\_name', 'last\_name'),

)

class PlayerClubHistory(db.Model): id = db.Column(db.Integer, primary\_key=True) player\_id = db.Column(db.Integer, db.ForeignKey('player\_profile.id'), nullable=False) club\_id = db.Column(db.Integer, db.ForeignKey('club.id'), nullable=False) start\_date = db.Column(db.Date, nullable=False) end\_date = db.Column(db.Date) player = db.relationship('PlayerProfile', back\_populates='history') club = db.relationship('Club', back\_populates='history')

\_\_table\_args\_\_ = (

UniqueConstraint('player\_id', 'club\_id', 'start\_date', name='uq\_club\_history'),

)

class ScoutingReport(db.Model): id = db.Column(db.Integer, primary\_key=True) scout\_id = db.Column(db.Integer, db.ForeignKey('scout\_profile.id'), nullable=False) player\_id = db.Column(db.Integer, db.ForeignKey('player\_profile.id'), nullable=False) notes = db.Column(db.Text) rating = db.Column(db.Integer) date\_created = db.Column(db.DateTime, default=datetime.utcnow) scout = db.relationship('ScoutProfile', back\_populates='reports') player = db.relationship('PlayerProfile', back\_populates='reports')

\_\_table\_args\_\_ = (

UniqueConstraint('scout\_id', 'player\_id', 'date\_created', name='uq\_scouting\_entry'),

)

class Match(db.Model): id = db.Column(db.Integer, primary\_key=True) date = db.Column(db.Date, nullable=False) time = db.Column(db.Time) venue = db.Column(db.String(120)) home\_team = db.Column(db.String(120)) away\_team = db.Column(db.String(120)) score\_home = db.Column(db.Integer) score\_away = db.Column(db.Integer) player\_stats = db.relationship('PlayerMatchStat', back\_populates='match', cascade="all, delete-orphan") team\_stats = db.relationship('TeamMatchStat', back\_populates='match', cascade="all, delete-orphan")

\_\_table\_args\_\_ = (

Index('ix\_match\_date', 'date'),

)

class PlayerMatchStat(db.Model): id = db.Column(db.Integer, primary\_key=True) player\_id = db.Column(db.Integer, db.ForeignKey('player\_profile.id'), nullable=False) match\_id = db.Column(db.Integer, db.ForeignKey('match.id'), nullable=False) minutes = db.Column(db.Integer) short\_pass\_succ = db.Column(db.Integer) short\_pass\_fail = db.Column(db.Integer) long\_pass\_succ = db.Column(db.Integer) long\_pass\_fail = db.Column(db.Integer) dribbles\_succ = db.Column(db.Integer) dribbles\_fail = db.Column(db.Integer) shots\_on = db.Column(db.Integer) shots\_off = db.Column(db.Integer) clearances = db.Column(db.Integer) aerial\_won = db.Column(db.Integer) aerial\_lost = db.Column(db.Integer) fouls\_for = db.Column(db.Integer) fouls\_against = db.Column(db.Integer) yellow = db.Column(db.Integer) red = db.Column(db.Integer) saves = db.Column(db.Integer) player = db.relationship('PlayerProfile', back\_populates='stats') match = db.relationship('Match', back\_populates='player\_stats')

\_\_table\_args\_\_ = (

UniqueConstraint('player\_id', 'match\_id', name='uq\_player\_match'),

CheckConstraint('minutes BETWEEN 0 AND 120', name='ck\_valid\_minutes'),

)

class TeamMatchStat(db.Model): id = db.Column(db.Integer, primary\_key=True) match\_id = db.Column(db.Integer, db.ForeignKey('match.id'), nullable=False) team\_name = db.Column(db.String(120), nullable=False) short\_pass\_succ = db.Column(db.Integer) short\_pass\_fail = db.Column(db.Integer) long\_pass\_succ = db.Column(db.Integer) long\_pass\_fail = db.Column(db.Integer) dribbles\_succ = db.Column(db.Integer) dribbles\_fail = db.Column(db.Integer) shots\_on = db.Column(db.Integer) shots\_off = db.Column(db.Integer) clearances = db.Column(db.Integer) aerial\_won = db.Column(db.Integer) aerial\_lost = db.Column(db.Integer) fouls\_for = db.Column(db.Integer) fouls\_against = db.Column(db.Integer) yellow = db.Column(db.Integer) red = db.Column(db.Integer) saves = db.Column(db.Integer) match = db.relationship('Match', back\_populates='team\_stats')

\_\_table\_args\_\_ = (

UniqueConstraint('team\_name', 'match\_id', name='uq\_team\_match'),

)