```python
 1  import os
 2  from datetime import datetime, timedelta, timezone
 3  from jose import jwt, JWTError
 4  from passlib.context import CryptContext
 5  from fastapi import Depends, HTTPException, status
 6  from fastapi.security import OAuth2PasswordBearer
 7  from app.repositories.csv_repository import CSVRepository
 8  from app.services.user_service import CSVUserService
 9
10  #  Password hashing
11  pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto")
12  bcrypt_context = pwd_context
13
14  # JWT setup
15  SECRET_KEY = os.getenv("AUTH_SECRET_KEY", "change-me")
16  ALGORITHM = os.getenv("AUTH_ALGORITHM", "HS256")
17  ACCESS_MINUTES = int(os.getenv("TOKEN_EXPIRE_MINUTES", "60"))
18
19  def create_access_token(*, username: str, user_id: int, is_admin: bool, minutes: int = ACCESS_MINUTES) -> str:
20      now = datetime.now(timezone.utc)
21      exp = now + timedelta(minutes=minutes)
22      payload = {"sub": username, "id": user_id,"is_admin": is_admin, "iat": int(now.timestamp()), "exp": int(exp.timestamp())
23      return jwt.encode(payload, SECRET_KEY, algorithm=ALGORITHM)
24
25  def decode_token(token: str):
26      try:
27          return jwt.decode(token, SECRET_KEY, algorithms=[ALGORITHM])
28      except JWTError:
29          raise HTTPException(
30              status_code=status.HTTP_401_UNAUTHORIZED,
31              detail="Invalid token",
32          )
33
34
35  def get_user_service() -> CSVUserService:
36      return CSVUserService(CSVRepository())
37
38  oauth2_scheme = OAuth2PasswordBearer(tokenUrl="/auth/token")
39
40  def get_current_user(token: str = Depends(oauth2_scheme),svc: CSVUserService = Depends(get_user_service),):
41      try:
42          payload = decode_token(token)
43          username = payload.get("sub")
44          user_id = payload.get("id")
45          if not username or user_id is None:
46              raise ValueError("missing_claims")
47      except Exception:
48          raise HTTPException(status_code=status.HTTP_401_UNAUTHORIZED, detail="invalid_token")
49
50      user = svc.get_by_username(username)
51      if not user or int(user["id"]) != int(user_id):
52          raise HTTPException(status_code=status.HTTP_401_UNAUTHORIZED, detail="user_not_found")
53      return user
```