Coverage for **app/deps.py**: 49%

41 statements | 20 run | 21 missing | 0 excluded

 coverage.py v7.11.3, created at 2025-11-20 19:17 -0800

```python
import os
from datetime import datetime, timedelta, timezone
from jose import jwt, JWTError
from passlib.context import CryptContext
from fastapi import Depends, HTTPException, status
from fastapi.security import OAuth2PasswordBearer
from app.utils.data_manager import CSVRepository
from app.services.user_service import CSVUserService

#  Password hashing
pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto")
bcrypt_context = pwd_context

# JWT setup
SECRET_KEY = os.getenv("AUTH_SECRET_KEY", "change-me")
ALGORITHM = os.getenv("AUTH_ALGORITHM", "HS256")
ACCESS_MINUTES = int(os.getenv("TOKEN_EXPIRE_MINUTES", "60"))

def create_access_token(*, username: str, user_id: int, minutes: int = ACCESS_MINUTES) -> str:
    now = datetime.now(timezone.utc)
    exp = now + timedelta(minutes=minutes)
    payload = {"sub": username, "id": user_id, "iat": int(now.timestamp()), "exp": int(exp.timestamp())}
    return jwt.encode(payload, SECRET_KEY, algorithm=ALGORITHM)

def decode_token(token: str):
    try:
        return jwt.decode(token, SECRET_KEY, algorithms=[ALGORITHM])
    except JWTError:
        raise HTTPException(status_code=status.HTTP_401_UNAUTHORIZED, detail="Invalid token")

# Repo + service singletons
_repo = CSVRepository()
_user_service = CSVUserService(_repo)

```

```
35 def get_user_service() -> CSVUserService:
36     return _user_service
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
```