```
from functools import wraps
import jwt
from flask import request
from backend import conn, config
import ibm_db

# Middleware function that checks for JWT token in header
# All routes that have the @token_required decorator will be protected



def token_required(f):
    @wraps(f)
    def decorated(*args, **kwargs):
        token = None
        if "Authorization" in request.headers:
            token = request.headers["Authorization"].split(" ")[1]
        if not token:
            return {
                "error": "Unauthorized"
            }, 401
        try:
            # Get the user's email from the decoded token
            data = jwt.decode(
                token, config["APP_SECRET"], algorithms=["HS256"])


            # Retreive user's info from the database
            sql = f"select * from users where email='{data['email']}'"
            stmt = ibm_db.prepare(conn, sql)
            ibm_db.execute(stmt)
            current_user = ibm_db.fetch_assoc(stmt)
```

```
        # If user does not exist throw error.
        if current_user is None:
            return {
                "error": "Unauthorized"
            }, 401
    except Exception as e:
        return {
            "error": str(e)
        }, 500

    # Pass the authorized user in function args.
    return f(current_user, *args, **kwargs)

return decorated
```