BugCatcher Report on Project: OWASPtest

Introduction

We've performed static analysis tests on your codebase. This reports highlights the potential issues that were found after running these tests on your codebase. Note that static analysis can only flag sections of code for review. Static analysis tests do not confirm the existence of any given vulnerability! A developer with security expertise can use this report to assess which sections of the codebase are most likely to have potential issues, and to give those sections special attention during a code review. Please use our "Annotation" feature to make notes about which issues require fixes, and which error messages are not applicable and can be safely ignored

High Severity

These are the highest severity issues that can be detected by our tool. When present, high severity issues could have a significant impact on the security of your codebase.

start_process_with_a_shell

Starting a process with a shell, possible injection detected, security issue.

File: models/Benefits/benefits.py {instance.start_line} - {instance.end_line}

blacklist

The pyCrypto library and its module PBKDF2 are no longer actively maintained and have been deprecated. Consider using pyca/cryptography library.

File: tests/views/test_users_pay.py {instance.start_line} - {instance.end_line}
File: models/utils.py {instance.start_line} - {instance.end_line}
File: views/api/users/views.py {instance.start_line} - {instance.end_line}
File: tests/models/test_model_work_info.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: tests/models/test_model_key_management.py {instance.start_line} - {instance.end_line}
File: tests/models/test_model_pay.py {instance.start_line} - {instance.end_line}
File: tests/vulnerabilities/test_a6_sensitive_data_exposure.py {instance.start_line} - {instance.end_line}
File: management/commands/seed.py {instance.start_line} - {instance.end_line}

Medium Severity

Medium severity issues can potentially be exploited by third parties, and are recommended to fix

blacklist

Use of possibly insecure function - consider using safer ast.literal_eval.

File: views/api/mobile/views.py {instance.start_line} - {instance.end_line}
File: views/api/mobile/views.py {instance.start_line} - {instance.end_line}

blacklist

Use of mark_safe() may expose cross-site scripting vulnerabilities and should be reviewed.

File: models/Message/message.py {instance.start_line} - {instance.end_line}
File: models/User/user.py {instance.start_line} - {instance.end_line}

django_mark_safe

Potential XSS on mark_safe function.

File: models/User/user.py {instance.start_line} - {instance.end_line}

blacklist
Pickle and modules that wrap it can be unsafe when used to deserialize untrusted data, possible security issue.

File: views/password_resets/views.py {instance.start_line} - {instance.end_line}

blacklist
Use of insecure MD2, MD4, MD5, or SHA1 hash function.

File: models/User/user.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: tests/views/test_password_reset.py {instance.start_line} - {instance.end_line}
File: models/User/user.py {instance.start_line} - {instance.end_line}
File: views/api/users/views.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: models/User/user.py {instance.start_line} - {instance.end_line}
File: tests/views/test_users.py {instance.start_line} - {instance.end_line}

hardcoded_sql_expressions
Possible SQL injection vector through string-based query construction.

File: models/Analytics/analytics.py {instance.start_line} - {instance.end_line}
File: views/users/views.py {instance.start_line} - {instance.end_line}

Low Severity
Low severity issues may violate best practices but be difficult to exploit. Formatting issues and other non-security concerns also are in this category

blacklist
Consider possible security implications associated with pickle module.

File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: tests/vulnerabilities/test_extra_remote_code_execution.py {instance.start_line} - {instance.end_line}
File: tests/views/test_password_reset.py {instance.start_line} - {instance.end_line}

hardcoded_password_string
Possible hardcoded password: ''

File: views/admin/views.py {instance.start_line} - {instance.end_line}

blacklist
Consider possible security implications associated with subprocess module.

File: tests/vulnerabilities/test_extra_remote_code_execution.py {instance.start_line} - {instance.end_line}

blacklist
Standard pseudo-random generators are not suitable for security/cryptographic purposes.

File: models/User/user.py {instance.start_line} - {instance.end_line}
File: models/User/user.py {instance.start_line} - {instance.end_line}

hardcoded_password_funcarg
Possible hardcoded password: 'ownerpass'

File: tests/mixins/auth_route_test_mixin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_user_benefit_forms.py {instance.start_line} - {instance.end_line}
File: tests/mixins/auth_route_test_mixin.py {instance.start_line} - {instance.end_line}
File: tests/vulnerabilities/test_a2_credential_enumeration.py {instance.start_line} - {instance.end_line}
File: tests/views/test_sessions.py {instance.start_line} - {instance.end_line}
File: tests/views/test_admin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_admin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_user_benefit_forms.py {instance.start_line} - {instance.end_line}
File: tests/views/test_password_reset.py {instance.start_line} - {instance.end_line}
File: tests/views/test_user_benefit_forms.py {instance.start_line} - {instance.end_line}
File: tests/views/test_users.py {instance.start_line} - {instance.end_line}
File: tests/mixins/auth_route_test_mixin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_users.py {instance.start_line} - {instance.end_line}
File: tests/mixins/auth_route_test_mixin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_admin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_user_benefit_forms.py {instance.start_line} - {instance.end_line}
File: tests/vulnerabilities/test_a2_credential_enumeration.py {instance.start_line} - {instance.end_line}
File: tests/views/test_admin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_admin.py {instance.start_line} - {instance.end_line}
File: tests/views/test_password_reset.py {instance.start_line} - {instance.end_line}

try_except_pass
Try, Except, Pass detected.

File: tests/vulnerabilities/test_extra_remote_code_execution.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: views/password_resets/views.py {instance.start_line} - {instance.end_line}
File: views/dashboard/views.py {instance.start_line} - {instance.end_line}

Informational
These items are purely informational items about best practices and are not security issues.

No items of this type were found by our scan.