# Test Cases Report - Notes Platform

## 1. White-box Testing

• Εκτελούνται όλες οι συνθήκες (if) για εγγραφή, login, προβολή σημειώσεων, πληρωμή.  
• Καλύπτονται όλα τα πιθανά μονοπάτια από αναζήτηση ως πληρωμή.

## 2. Basic Path Testing

• Υπολογισμός Κυκλωματικής Πολυπλοκότητας V(G) = P + 1 ≈ 7 + 1 = 8  
• Εξετάζονται μονοπάτια όπως: επιτυχής login, αποτυχία login, φίλτρα επί πληρωμή/δωρεάν, πληρωμή επιτυχής/αποτυχής.

## 3. Equivalence Partitioning & Boundary Value Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Περιγραφή | Είσοδος | Αναμενόμενο |
| TC1 | Εγγραφή με άκυρο email | email = "wrong@outlook.com" | Σφάλμα |
| TC2 | Εγγραφή με λάθος ρόλο | role = "admin" | Σφάλμα |
| TC3 | Login με σωστά στοιχεία | email = "student@gmail.com" | Επιτυχής είσοδος |
| TC4 | Αποτυχία login | email = "fake@gmail.com" | Μήνυμα λάθους |
| TC5 | Αναζήτηση χωρίς σημειώσεις | Καμία | Μήνυμα: Δεν βρέθηκαν σημειώσεις |
| TC6 | Εμφάνιση δωρεάν σημειώσεων | is\_paid = 0 | Προβολή με messagebox |
| TC7 | Πληρωμή σημείωσης με επιτυχία | is\_paid = 1 | Επιτυχής πληρωμή |

## 4. Κώδικας Ελέγχου (Test Code)

import unittest

from sqlalchemy import create\_engine

from sqlalchemy.orm import sessionmaker

from shmeioseis import Base, User, Course, Note, hash\_password, App

import tkinter as tk

class TestNoteAccess(unittest.TestCase):

    @classmethod

    def setUpClass(cls):

        # Σύνδεση με πραγματική βάση

        cls.engine = create\_engine("sqlite:///study\_platform.db")

        Base.metadata.create\_all(cls.engine)

        cls.Session = sessionmaker(bind=cls.engine)

        cls.session = cls.Session()

        # Καθαρισμός προηγούμενων δεδομένων (προαιρετικό)

        cls.session.query(User).filter\_by(email="teststudent@gmail.com").delete()

        cls.session.query(Course).filter\_by(title="Δοκιμαστικό Μάθημα").delete()

        cls.session.commit()

        # Προσθήκη test χρήστη

        cls.test\_user = User(

            first\_name="Test",

            last\_name="Student",

            email="teststudent@gmail.com",

            password=hash\_password("testpass"),

            role="student",

            department\_id=None

        )

        cls.session.add(cls.test\_user)

        cls.session.commit()

        # Προσθήκη μαθήματος

        cls.test\_course = Course(title="Δοκιμαστικό Μάθημα")

        cls.session.add(cls.test\_course)

        cls.session.commit()

        # Προσθήκη σημείωσης

        cls.test\_note = Note(

            uploader\_id=cls.test\_user.id,

            course\_id=cls.test\_course.id,

            title="Σημείωση Test",

            file\_path="path/to/fake\_note.pdf",

            is\_paid=0

        )

        cls.session.add(cls.test\_note)

        cls.session.commit()

    def setUp(self):

        self.root = tk.Tk()

        self.app = App(self.root)

        # Εισαγωγή credentials

        self.app.email\_entry.insert(0, "teststudent@gmail.com")

        self.app.pass\_entry.insert(0, "testpass")

    def test\_login\_and\_course\_display(self):

        self.app.login()

        # Έλεγχος ετικέτας welcome

        welcome\_found = any(

            isinstance(w, tk.Label) and "Welcome Test Student" in w.cget("text")

            for w in self.root.winfo\_children()

        )

        self.assertTrue(welcome\_found, "Login failed or welcome not shown")

        # Προσομοίωση αναζήτησης μαθημάτων

        self.app.search\_notes()

        course\_titles = [

            w.cget("text") for w in self.app.course\_buttons\_frame.winfo\_children()

            if isinstance(w, tk.Button)

        ]

        self.assertIn("Δοκιμαστικό Μάθημα", course\_titles, "Course not displayed in search")

    def tearDown(self):

        self.root.destroy()

    @classmethod

    def tearDownClass(cls):

        cls.session.query(Note).filter\_by(title="Σημείωση Test").delete()

        cls.session.query(Course).filter\_by(title="Δοκιμαστικό Μάθημα").delete()

        cls.session.query(User).filter\_by(email="teststudent@gmail.com").delete()

        cls.session.commit()

        cls.session.close()

        cls.engine.dispose()

if \_\_name\_\_ == "\_\_main\_\_":

    unittest.main()

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, εικονίδιο υπολογιστή

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, λογισμικό, εικονίδιο υπολογιστή, λογισμικό πολυμέσων

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, λογισμικό, πολυμέσα, εικονίδιο υπολογιστή

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, γραμματοσειρά, στιγμιότυπο οθόνης, γραμμή

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, λογισμικό, στιγμιότυπο οθόνης, εικονίδιο υπολογιστή

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.

Εικόνα που περιέχει κείμενο, λογισμικό, πολυμέσα, λειτουργικό σύστημα

Το περιεχόμενο που δημιουργείται από τεχνολογία AI ενδέχεται να είναι εσφαλμένο.