6510615211 พรนัชชา ประทีปสังคม

6510685057 ญาณัชทัฬห์ คงกระจ่าง

Test Method

1. app_courses

1.1 Class SubjectModelTest(TestCase) : ทดสอบการทำงาน model class ซึ่งที่ตั้งไว้ในโปรแกรมนี้ คือ class ชื่อว่า Subject

1.1.1 setUp(self) : เป็นฟังก์ชันที่ใช้เพื่อสร้างข้อมูลตัวอย่างในการทดสอบฟังก์ชัน ต่อๆไปอีกที ในโปรแกรมกำหนดค่าตามนี้

- self.subject = Subject.objects.create(
  course_id="CN202",
  course_name="Data algorithms 1",
  course_semester="1/2567",
  course_amount=9999,
  course_status=Subject.CourseStatus.AVAILABLE,)
)

1.1.2 test_subject_create(self) : เอาไว้ตรวจสอบว่าข้อมูลที่สร้างขึ้น มีค่าตรงกับค่าที่กำหนดมั้ย (เทียบค่า self.subject.course_id กับค่าที่สร้าง)

1.1.3 test _subject_string_representation(self) :
ตรวจสอบว่าสามารถส่งคืนรายละเอียดวิชาต่างๆได้ถูกต้อง (เทียบค่า str(self.subject) กับค่าที่ str ที่เราต้องการ)

1.1.4 test_update_course_status_and_amount : เอาไว้ทดสอบว่าสถานะกับจำนวนที่เราตั้งไว้ มันมีการเปลี่ยนแปลงสถานะมั้ย แล้วเรียก save() เพื่อตรวจสอบว่าค่าที่มันเปลี่ยนแปลงได้อัปเดตลงฐานข้อมูลหรือไม่

1.1.5 test_course_amount_never_nagative(self) : เอาไว้ตรวจสอบว่าcourse amount ที่ตั้งไว้จะไม่มีวันมีค่าติดลบได้

1.2 Class StudentModelTest(TestCase)

1.2.1 setUp(self) : เป็นฟังก์ชันที่ใช้เพื่อสร้างข้อมูลตัวอย่างในการทดสอบฟังก์ชันต่อๆไป อีกที ใน class นี้มันจะสร้าง Subject กับ Student

```
self.subject = Subject.objects.create(
        course_id="CN202",
        course_name="Data algorithm 1",
        course_semester="1/2567",
        course_amount=9999,
        course_status=Subject.CourseStatus.AVAILABLE,)
self.student = Student.objects.create(
        student_id="6510681234",
        first_name="likestudy",
```

```
            last_name="makmak",
            email="likestudy.mak@dome.tu.ac.th"
        )
        self.student.enrolled_subjects.add(self.subject)
```

1.2.2 test_student_create(self) : เอาไว้ตรวจสอบว่าข้อมูลที่สร้างขึ้น มีค่าตรงกับค่าที่กำหนดมั้ย เช่น student Id กับemail ค่าตรงที่ตั้งไว้มั้ย

1.2.3 test_student_enrollment(self) : ไว้ทดสอบการลงทะเบียนของนักศึกษาและวิชา เพื่อตรวจสอบความถูกต้องของการลงทะเบียน

1.2.4 test_remove_student_enrollment(self) : ไว้ทดสอบการลบนักศึกษาจากการลงทะเบียนแต่ละวิชาว่าดำเนินการได้ถูกต้องหรือไม่

1.2.5 test_duplicate _id(self) : ถ้ามีการซ้ำกันของ Id นักศึกษา มันควรจะเกิด error เนื่องจากมันเป็นkey ที่ไม่ควรซ้ำกัน

1.3 Class CourseViewTests(TestCase)

1.3.1 setUp(self) : เป็นฟังก์ชันที่ใช้เพื่อสร้างข้อมูลตัวอย่างในการทดสอบฟังก์ชันต่อๆไป อีกที ในclass นี้ มันจะสร้าง Subject ตัวอย่างเอาไว้ทดสอบ view ที่เกี่ยวข้องกับข้อมูลของcourse

```
self.client = Client()
self.subject = Subject.objects.create(
        course_id="CN202",
        course_name="Data algorithm 1",
        course_semester="1/2567",
        course_amount=9999,
        course_status=Subject.CourseStatus.AVAILABLE,)
```

1.3.2 test_courses_view_get(self) : ใช้เรียกเพื่อดูหน้าที่แสดง course ด้วยการเรียก HTTP GET เพื่อตรวจสอบว่าดึง courses.html ถูกต้อง

1.3.3 test_course_view_post_valid_data(self) : ทดสอบการลงทะเบียนผ่าน HTTP POSTว่ามีการอัปเดตข้อมูลถูกต้องหรือไม่

1.3.5 test_enroll_check_view(self) : ไว้ตรวจสอบว่าเมื่อเรียกใช้ http get จะต้องได้หน้าตรวจสอบการลงทะเบียนแบบที่เราต้องการ

1.3.6 test_subject _transition(self) : ไว้ทดสอบการเปลี่ยนสถานะของการลงทะเบียน พวก status , amount ต่างๆ

2. app_registation : การเขียน test ใน app นี้ ส่วนใหญ่จะเกี่ยวข้องกับพวก view เนื่องจาก ส่วนใหญ่เป็นการแสดงผลหน้าแอป ซึ่งเราจะ test ว่า view ต่างๆทำงานได้ถูกต้อง

2.1 ViewsTestCase Class(TestCase) : เป็น class ทดสอบของ Django ที่ใช้ในการทดสอบการทำงานของ view

2.1.1 setUp(self) : เป็นฟังก์ชันที่ใช้เพื่อสร้างข้อมูลตัวอย่างในการทดสอบฟังก์ชัน ต่อๆไปอีกทีในclassนี้ จะเตรียม self.student_data ที่ใช้ในการทดสอบ view ที่เกี่ยวกับการลงทะเบียนหลักๆ ในโปรแกรมนี้เขียนไว้ว่า

```
self.student_data = {
```

'student_id': '6510681234',
'name': 'likestudy makmak',}

2.1.2 test_home_view(self) : ใช้ทดสอบว่าหน้า home view ทำงานได้ถูกต้อง

2.1.3 test_about_view(self) : ใช้ทดสอบว่าหน้า about view ทำงานได้ถูกต้อง

2.1.4 test_enroll_check_view(self) : ใช้ทดสอบว่าหน้า การตรวจสอบการลงทะเบียนทำงานได้ถูกต้อง

3. app_user : การเขียน test ใน app นี้ จะใช้เพื่อทดสอบเกี่ยวกับ user เป็นหลัก พวกบัญชีผู้ใช้ รหัสผ่าน เป็นต้น

3.1 class UserModelTest(TestCase) : ทดสอบการทำงานของฟังก์ชัน User

3.1.1 setUp(self) : เป็นฟังก์ชันที่ใช้เพื่อสร้างข้อมูลตัวอย่างในการทดสอบฟังก์ชัน ต่อๆไปอีกที ใน class นี้จะสร้างชื่อผู้ใช้และรหัสผ่าน ขึ้นมา

```
self.user = User.objects.create_user(
        username='testuser',
        password='testpassword'
)
```

3.1.2 test_user_create(self) : ไว้ใช้ทดสอบการสร้างชื่อผู้ใช้ถูกต้องหรือไม่

3.1.3 test_user_password(self) : ไว้ใช้ทดสอบการเข้ารหัสผ่านผู้ใช้ถูกต้องหรือไม่

- self.assertNotEqual(self.user.password, 'testpassword') *รหัสที่กำหนดไว้ในโค้ด

: ตรวจสอบว่ารหัสที่เก็บในฐานข้อมูลไม่ใช่รหัสเดิม ถ้าซ้ำจะเกิด error

3.1.4 test_uniqe_username(self) : ไว้ตรวจสอบว่าสร้างชื่อผู้ใช้ไม่ซ้ำกัน ถ้าซ้ำจะเกิด error

3.1.5 test_str_method(self) : ไว้ทดสอบว่า __str__(self) method ของ model User คืนค่าถูกต้อง