



TEST UNITARIO: TRABAJO PRÁCTICO

IVANA EBRI

METODOLOGÍA DE PRUEBAS – 2DO CUATRIMESTRE 2023



SISTEMA A TESTEAR



TEST REALIZADOS Y RESULTADOS ESPERADOS

TEST CASE

Nro TC	TC-UT-01
Nombre de TC	Conexión a base de datos
Descripción	Verificar que la conexión a la base de datos se realiza correctamente
Supuestos y condiciones previas	La aplicación se encuentra instanciada y se ejecuta la conexión a la base
Datos de prueba	-
Pasos a ejecutar	Instanciar y ejecutar la clase TestAppEmergenciaConexion
Resultado esperado	La base de datos está conectada
Resultado real/condiciones posteriores	Se ha conectado a la base de datos Se cierra la conexión
Estado de prueba	APROBADO

```
class TestAppEmergenciaConexion(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
        self.app._conectar_db()

    def tearDown(self):
        self.app._cerrar_db()
        sqlite_db.close()

    def test_conectar_db(self):
        """Verificar que la conexión a la base de
        datos se realiza correctamente"""
        self.assertTrue(sqlite_db.connect, "La
        conexion no se realizó")
```

Nro TC	TC-UT-02
Nombre de TC	Desconexión a base de datos
Descripción	Verificar que se cierra correctamente la conexión a la base de datos
Supuestos y condiciones previas	La aplicación se encuentra instanciada, se intenta cerrar la conexión, por si estuviese abierta, y se ejecuta la conexión a la base
Datos de prueba	-
Pasos a ejecutar	Instanciar y ejecutar la clase TestAppEmergenciaCierre
Resultado esperado	La base de datos está desconectada
Resultado real/condiciones posteriores	Se ha desconectado la base de datos
Estado de prueba	APROBADO

```
Class TestAppEmergenciaCierre(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
        self.app._cerrar_db()
        self.app._conectar_db()

    def test_cerrar_conexion(self):
        """Verificar que la conexión a la base de
        datos se cierra correctamente"""
        self.app._cerrar_db()
        self.assertTrue(sqlite_db.is_closed(), "La
        conexion no se encuentra cerrada")
```


Nro TC	TC-UT-03
Nombre de TC	Emergencias solicitadas
Descripción	Se verifica si lee y devuelve de manera correcta la cantidad de emergencias solicitadas
Supuestos y condiciones previas	Se inicializo la aplicación
Datos de prueba	Cantidad de emergencias = 2
Pasos a ejecutar	Instanciar la clase TestAppEmergenciaMethod Ejecutar el método test_ultimas_emergencias
Resultado esperado	Devuelve una lista con 2 elementos (las dos emergencias solicitadas)
Resultado real/condiciones posteriores	La longitud de la lista es de 2
Estado de prueba	APROBADO

Nro TC	TC-UT-04
Nombre de TC	Verificar coordenadas con direccion
Descripción	Se verifica que la función de rastreo de dirección este devolviendo la dirección correcta
Supuestos y condiciones previas	Se inicializo la aplicación
Datos de prueba	Ubicación 1 = -34.763704,-58.208684 Ubicación 2 = -34.544762,-58.450766
Pasos a ejecutar	Instanciar la clase TestAppEmergenciaMethod Ejecutar el método test_ciudad_cercana
Resultado esperado	Retorna las direcciones correctas a partir de las coordenadas enviadas
Resultado real/condiciones posteriores	Retorna "Diagonal Lisandro de la Torre, Berazategui, Partido de Berazategui, Buenos Aires, B1880BFG, Argentina" y "21642, Conector Interciclovías, Barrio River, Belgrano, Buenos Aires, Comuna 13, Ciudad Autónoma de Buenos Aires, C1424BCL, Argentina"
Estado de prueba	APROBADO

```
class
TestAppEmergenciaMethods(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
    def test_ultimas_emergencias(self):
        """Verificar si se devuelven las últimas
2 emergencias correctamente"""
        emergencias=self.app._ultimas_emergencia
s(2)
        cant_registros = len(emergencias)
        self.assertEqual(cant_registros, 2)
```

```
class TestAppEmergenciaMethods(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
    def test_ciudad_cercana(self):
        """Verificar si la función ciudad_cercana devuelve la ubicación
esperada"""
        berazategui = self.app.ciudad_cercana(-34.763704,-58.208684)
        river= self.app.ciudad_cercana(-34.544762,-58.450766)
        self.assertEqual(berazategui.address,"Diagonal Lisandro de la Torre,
Berazategui, Partido de Berazategui, Buenos Aires, B1880BFG, Argentina")
        self.assertEqual(river.address, "21642, Conector Interciclovías,
Barrio River, Belgrano, Buenos Aires, Comuna 13, Ciudad Autónoma de Buenos
Aires, C1424BCL, Argentina")
```

Nro TC	TC-UT-05
Nombre de TC	Existencia de tablas de bd
Descripción	Comprueba que las tablas que componen la base de datos existan
Supuestos y condiciones previas	Se inicializo la aplicación
Datos de prueba	-
Pasos a ejecutar	Instanciar la clase TestAppEmergenciaMethod Ejecutar el método test_mapear_orm
Resultado esperado	Las tres tablas existen
Resultado real/condiciones posteriores	Las 3 tablas de la bd existen
Estado de prueba	APROBADO

```
class TestAppEmergenciaMethods(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
    def test_mapear_orm(self):
        """Verificar que la función _mapear_orm crea
las tablas correctamente"""
        self.app._mapear_orm()
        #verifico si existen todas
        self.assertTrue(AreaEmergencia.table_exists(
))
        self.assertTrue(TipoEmergencia.table_exists(
))
        self.assertTrue(RegistroEmergencias.table_ex
ists())
```

Nro TC	TC-UT-06
Nombre de TC	Comprobación de desplegables
Descripción	Las tablas que componen los desplegables tienen mas de 0 registros y contienen la cantidad de opciones de acuerdo a las reglas de negocio establecidas
Supuestos y condiciones previas	Se inicializo la aplicación
Datos de prueba	-
Pasos a ejecutar	Instanciar la clase TestAppEmergenciaMethod Ejecutar el método test_datos_desplegables
Resultado esperado	Las tablas no están vacías y contienen: <ul style="list-style-type: none"> - Áreas 7 registros - Tipo 5 registros
Resultado real/condiciones posteriores	Áreas no contiene 0 registros Áreas contine 7 registros Tipo no contiene 0 registros Tipo contine 5 registros
Estado de prueba	APROBADO

```
class TestAppEmergenciaMethods(unittest.TestCase):
    def setUp(self):
        self.app = App_emergencia()
    def test_datos_desplegables(self):
        """Verificar que las tablas de Área y Tipo
de emergencia no se encuentran vacías"""
        areas_count =
AreaEmergencia.select().count()
        tipos_count =
TipoEmergencia.select().count()
        # Realizar las aserciones
        self.assertNotEqual(areas_count, 0)
        self.assertEqual(areas_count, 7)
        self.assertNotEqual(tipos_count, 0)
        self.assertEqual(tipos_count, 5)
```




MUCHAS
GRACIAS

ALGUIEN@EJEMPLO.COM