CODE:

class Investor:

    def \_\_init\_\_(self, name, available\_funds):

        self.name = name

        self.available\_funds = available\_funds

class FilmProject:

    def \_\_init\_\_(self, project\_id, project\_name, required\_funds):

        self.project\_id = project\_id

        self.project\_name = project\_name

        self.required\_funds = required\_funds

        self.current\_funds = 0

    def add\_funds(self, amount):

        self.current\_funds += amount

        return self.current\_funds >= self.required\_funds  # Returns True if fully funded

class FundingRequest:

    def \_\_init\_\_(self, request\_id, project, amount):

        self.request\_id = request\_id

        self.project = project

        self.amount = amount

        self.status = 'open'  # Possible statuses: open, funded, cancelled

class FundingPlatform:

    def \_\_init\_\_(self):

        self.projects = {}

        self.investors = {}

        self.requests = {}

        self.transactions = []

    def add\_project(self, project\_id, project\_name, required\_funds):

        self.projects[project\_id] = FilmProject(project\_id, project\_name, required\_funds)

    def remove\_project(self, project\_id):

        if project\_id in self.projects:

            del self.projects[project\_id]

    def add\_investor(self, investor\_name, funds):

        new\_investor = Investor(investor\_name, funds)

        self.investors[investor\_name] = new\_investor

    def remove\_investor(self, investor\_name):

        if investor\_name in self.investors:

            del self.investors[investor\_name]

    def create\_request(self, project\_id, amount):

        project = self.projects.get(project\_id)

        if project:

            request = FundingRequest(len(self.requests) + 1, project, amount)

            self.requests[request.request\_id] = request

            return request.request\_id

    def cancel\_request(self, request\_id):

        if request\_id in self.requests:

            self.requests[request\_id].status = 'cancelled'

    def connect\_filmmakers\_with\_investors(self, project\_id):

        project = self.projects.get(project\_id)

        if not project:

            return []

        # Match investors based on available funds and project fund needs

        potential\_investors = [

            inv for inv in self.investors.values() if inv.available\_funds >= project.required\_funds - project.current\_funds]

        return potential\_investors

    def manage\_funding\_transactions(self, transaction\_data):

        project\_id, investor\_name, amount = transaction\_data

        project = self.projects.get(project\_id)

        investor = self.investors.get(investor\_name)

        if project and investor and investor.available\_funds >= amount:

            project.add\_funds(amount)

            investor.available\_funds -= amount

            self.transactions.append(transaction\_data)

            return True

        return False

### Unit Tests Using Python's unittest

#Now, let's write some basic unit tests:

import unittest

class TestFundingPlatform(unittest.TestCase):

    def setUp(self):

        self.platform = FundingPlatform()

        self.platform.add\_project(1, "Epic Space Opera", 100000)

        self.platform.add\_investor("Alice", 50000)

        self.platform.add\_investor("Bob", 75000)

    def test\_project\_addition(self):

        self.assertIn(1, self.platform.projects)

    def test\_investor\_addition(self):

        self.assertIn("Alice", self.platform.investors)

        self.assertIn("Bob", self.platform.investors)

    def test\_funding\_transaction(self):

        self.platform.manage\_funding\_transactions((1, "Alice", 50000))

        self.assertEqual(self.platform.projects[1].current\_funds, 50000)

    def test\_connect\_filmmakers\_with\_investors(self):

        self.platform.add\_investor("charlie",100000)

        potential\_investors = self.platform.connect\_filmmakers\_with\_investors(1)

        self.assertEqual(len(potential\_investors), 1)  # Both investors have sufficient funds

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()

OUTPUT :-

....

----------------------------------------------------------------------

Ran 4 tests in 0.001s

OK