```
from numpy import arange
from pylab import *


def trapazoidal(f, ax, bx, ay, by, h=0.001):
    h2 = h**2
    Nx = int(abs(bx - ax) / h)
    Ny = int(abs(by - ay) / h)
    corner = 0.25 * (f(ax, ay) + f(ax, by) + f(bx, ay) + f(bx, by))
    edge = 0.5 * (sum([f(ax, ay + i * h) for i in range(1, Ny)]) + sum([
        f(bx, ay + i * h) for i in range(1, Ny)
    ]) + sum([f(ax + i * h, ay) for i in range(1, Nx)]) + sum(
        [f(ax + i * h, by) for i in range(1, Nx)]))
    inner = sum([
        sum([f(ax + i * h, ay + k * h)
             for k in range(1, Ny)])
        for i in range(1, Nx)
    ])
    return (h**2) * (corner + edge + inner)

def p2():
    ep = 8.854187871e-12

    def dist(p1, p2):
        return sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2 +
                    (p1[2] - p2[2])**2)

    def diff(a, b):
        return abs(a - b) >= 1e-5

    def V(q, r):
        return q / (4 * pi * ep * r)

    def E(q, r):
        return q / (4 * pi * ep * r**2)

    def potential(x, y, z):
        return V(-1, dist([-0.05, 0, 0], [x, y, z])) + V(
            1, dist([0.05, 0, 0], [x, y, z]))

    def Ex(x, y, z):
        return (potential(x + 1e-8, y, z) - potential(x, y, z)) / 1e-8

    def Ey(x, y, z):
        return (potential(x, y + 1e-8, z) - potential(x, y, z)) / 1e-8

    def a():
        data = [[potential(x, y, 0)
                 for x in arange(-0.5, 0.5, 0.01)]
                for y in arange(-0.5, 0.5, 0.01)]
        imshow(data, vmax=1e10, vmin=-1e10)
        show()

    def b():
        vec_u = [[-Ex(x, y, 0)
                  for x in arange(-0.5, 0.5, 0.01)]
                 for y in arange(-0.5, 0.5, 0.01)]
        vec_v = [[-Ey(x, y, 0)
                  for x in arange(-0.5, 0.5, 0.01)]
                 for y in arange(-0.5, 0.5, 0.01)]
        for i in range(len(vec_u)):
            for j in range(len(vec_u[i])):
                if abs(vec_u[i][j]) > 1e12 or abs(vec_v[i][j]) > 1e12:
```

```python
                    vec_u[i][j] = 0
                    vec_v[i][j] = 0
        quiver(vec_u, vec_v)
        show()

    def c():
        L = .1
        sigma = lambda x, y, z: 100 * sin(2 * pi * x / L) * sin(2 * pi * y / L)

        def gen(a, b, c):
            return lambda x, y: 0 if (b-y) == (a-x) == 0 else (sigma(x, y, 0) / (4 * pi * ep * sqrt((a - x)**2 + (b - y)**2 + (c - 0)**2)))

        potential = [[ trapazoidal(gen(a, b, 0), -0.05, 0.05, -0.05, 0.05, 0.01) for a in arange(-0.05, 0.05, 0.01) ] for b in arange(-0.05, 0.05, 0.01)]

        vec_u = [[(potential[i + 1][j] - potential[i][j]) / 0.01 if i != (len(potential[j])) - 1 else (potential[i-1][j]-potential[i][j])/0.01
                  for i in range(len(potential[j]))]
                 for j in range(len(potential) - 1)]
        vec_v = [[(potential[i][j + 1] - potential[i][j]) / 0.01 if i != (len(potential[j])) - 1 else (potential[i-1][j]-potential[i][j])/0.01
                  for i in range(len(potential[j]))]
                 for j in range(len(potential) - 1)]
        imshow(potential)
        quiver(vec_u, vec_v)
        show()

    a()
    b()
    c()

if __name__ == "__main__":
    p2()
```