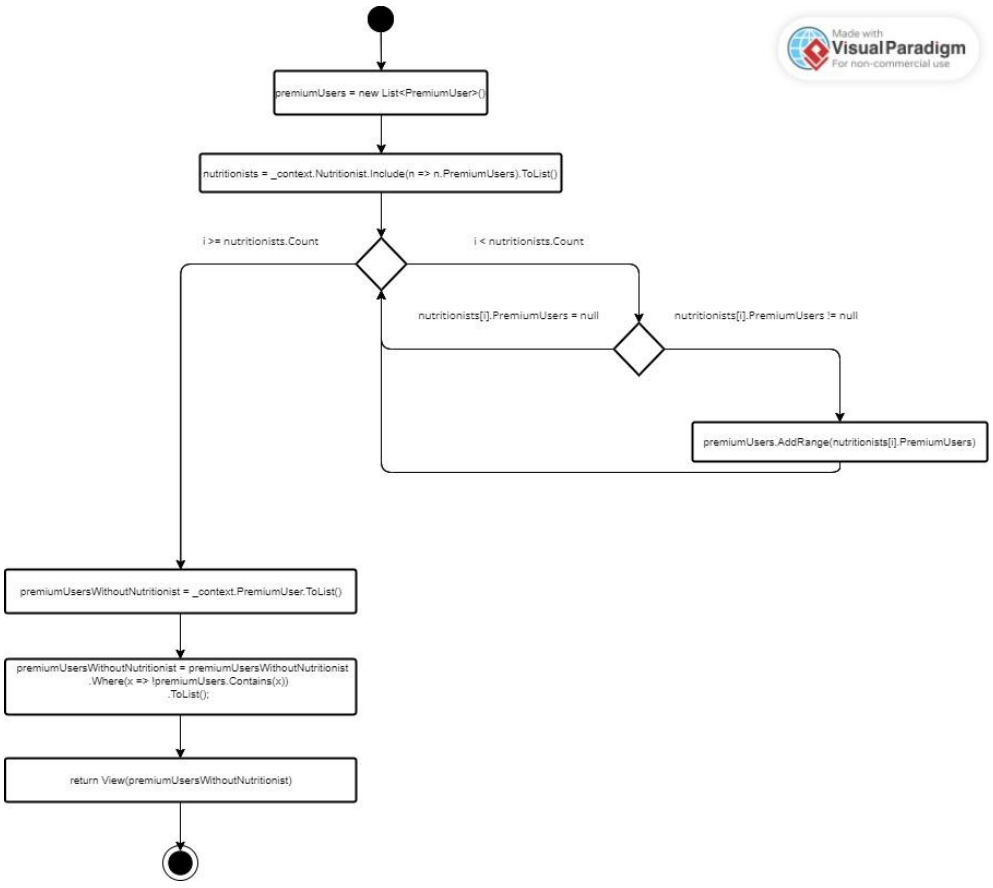
WHITEBOX TESTIRANJE – INDEX METODA ZA ADMINCONTROLLER

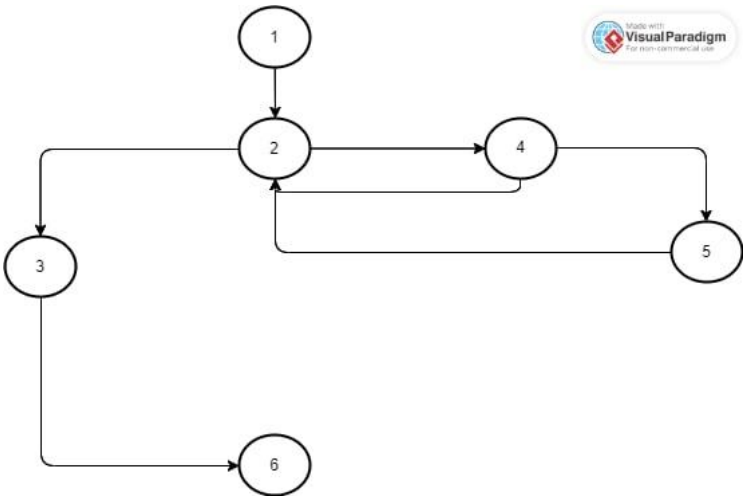
Metoda:

```
5 references | we ran into an exception loading metrics for this method - please contact support | 3/4 passing
public async Task<IActionResult> Index()
{
    // Retrieve premium users from the database
    var premiumUsers = new List<PremiumUser>(); // Create a list to hold premium users

    // Retrieve all nutritionists and their associated premium users
    var nutritionists = _context.Nutritionist.Include(n => n.PremiumUsers).ToList();

    // Loop through each nutritionist
    for (int i = 0; i < nutritionists.Count; i++)
    {
        // Check if the current nutritionist has associated premium users
        if (nutritionists[i].PremiumUsers != null)
        {
            // Add the premium users associated with this nutritionist to the premiumUsers
            premiumUsers.AddRange(nutritionists[i].PremiumUsers);
        }
    }

    // Retrieve all premium users from the database
    var premiumUsersWithoutNutritionist = _context.PremiumUser.ToList();

    // Filter out premium users that are associated with nutritionists
    premiumUsersWithoutNutritionist = premiumUsersWithoutNutritionist
        .Where(x => !premiumUsers.Contains(x))
        .ToList();

    // Pass the filtered premium users to the view
    return View(premiumUsersWithoutNutritionist);
}
```

Dijagram toka:



Graf programskog toka:



Putevi:

1-2-3-6

1-2-4-2-3-6

1-2-4-5-2-3-6

Za navedene puteve napisana su 3 testna slučaja:

1-2-3-6:

```
[TestMethod]
● | 0 references | we ran into an exception loading metrics for this method - please contact support
public async Task Index_ReturnsCorrectView_WhenNoNutritionists()
{

    var nutritionists = new List<Nutritionist>();

    var premiumuser1 = new PremiumUser();
    var premiumuser2 = new PremiumUser();
    var premiumuser3 = new PremiumUser();

    var premiumUsersWithoutNutritionist = new List<PremiumUser>
{
   premiumuser1,
   premiumuser2,
   premiumuser3
};

    _mockDbContext.Setup(c => c.Nutritionist)
      .ReturnsDbSet(nutritionists);

    _mockDbContext.Setup(c => c.PremiumUser)
        .ReturnsDbSet(premiumUsersWithoutNutritionist);

    var result = await _controller.Index();
    var viewResult = result as ViewResult;

    // Assert
    Assert.IsInstanceOfType(viewResult.Model, typeof(List<PremiumUser>));
    var model = viewResult.Model as List<PremiumUser>;

    Assert.AreEqual(3, model.Count());
}
```

1-2-4-2-3-6:

```
[TestMethod]
◆ | 0 references | we ran into an exception loading metrics for this method - please contact support
public async Task Index_ReturnsEmptyView_WhenNoPremiumUsers()
{

    var nutritionists = new List<Nutritionist>
{
    new Nutritionist { PremiumUsers = new List<PremiumUser>() }
};


    _mockDbContext.Setup(c => c.Nutritionist)
      .ReturnsDbSet(nutritionists);

    _mockDbContext.Setup(c => c.PremiumUser)
      .ReturnsDbSet(new List<PremiumUser>());

    // Act
    var result = await _controller.Index();

    // Assert
    var viewResult = result as ViewResult;
    Assert.IsTrue(viewResult.ViewData.Model == null || !((IEnumerable<PremiumUser>)viewResult.Model).Any());
    var model = viewResult.Model as List<PremiumUser>;

    Assert.AreEqual(0, model.Count());
}
```

1-2-4-5-2-3-6

```
[TestMethod]
◆ | 0 references | we ran into an exception loading metrics for this method - please contact support
public async Task Index_ReturnsFilteredPremiumUsers()
{
    var premiumuser1 = new PremiumUser();
    var premiumuser2 = new PremiumUser();
    var premiumuser3 = new PremiumUser();

    var nutritionists = new List<Nutritionist>
{
    new Nutritionist { PremiumUsers = new List<PremiumUser> { premiumuser1 } },
    new Nutritionist { PremiumUsers = new List<PremiumUser>() }
};

    var premiumUsersWithoutNutritionist = new List<PremiumUser>
{
   premiumuser1,
   premiumuser2,
   premiumuser3
};

    _mockDbContext.Setup(c => c.Nutritionist)
        .ReturnsDbSet(nutritionists);

    _mockDbContext.Setup(c => c.PremiumUser)
      .ReturnsDbSet(premiumUsersWithoutNutritionist);

    // Act
    var result = await _controller.Index();
    var viewResult = result as ViewResult;

    // Assert
    Assert.IsInstanceOfType(viewResult.Model, typeof(List<PremiumUser>));
    var model = viewResult.Model as List<PremiumUser>;

    Assert.AreEqual(2, model.Count()); // Verify the correct number of premium users are returned
}
```

Ovim testovima su obuhvaćeni svi putevi, petlje, grane te sve linije metode.