Muaz Sikirić 19171

# White box testiranje - izvještaj

Metoda *AddHavenCoins* u klasi *HavenCoinsService*:

```
public bool AddHavenCoins(int id, int coins)
{
    if (coins < 0)
    {
        throw new ArgumentException("Can't subtract coins");
    }
    try
    {
        var usersLine = csv.First(element => element[0] == id);
        if (usersLine[1] + coins > 50)
        {
            return false;
        }
        usersLine[1] += coins;
        SaveNewCsv();
        return true;
    }
    catch (InvalidOperationException)
    {
        throw new ArgumentException("User with such id doesn't
exist");
    }
}
```

Line coverage:

```
[TestMethod]
public void AddHavenCoins_ValidIdAndPositiveCoins_ReturnTrue()
{
    int id = 1;
    int coins = 28;
    int addCoins = 7;

    bool result = havenCoinsService.AddHavenCoins(id, addCoins);
    Assert.IsTrue(result, "AddHavenCoins must return true for valid
addition.");
    Assert.AreEqual(coins + addCoins,
havenCoinsService.getHavenCoins(id));
}

[TestMethod]
[ExpectedException(typeof(ArgumentException))]
public void AddHavenCoins_InvalidId_ShouldThrowArgumentException()
{
    int id = -1;
    int addCoins = 5;
    havenCoinsService.AddHavenCoins(id, addCoins);
}

[TestMethod]
[ExpectedException(typeof(ArgumentException))]
public void
AddHavenCoins_NegativeCoins_ShouldThrowArgumentException()
{
    int id = 1;
    int addCoins = -1;
    havenCoinsService.AddHavenCoins(id, addCoins);
}

[TestMethod]
public void AddHavenCoins_ExceedsMaximumCoins_ReturnFalse()
{
    int id = 1;
    int coins = 28;
    int addCoins = 23;

    bool result = havenCoinsService.AddHavenCoins(id, addCoins);

    Assert.IsFalse(result, "AddHavenCoins must return false for
exceeding the maximum (50) coins.");
    Assert.AreEqual(coins, havenCoinsService.getHavenCoins(id));
}
```