2016

Beveiligingsverslag DankBank

REMCO LA BRIJN

Inhoud

Inleiding

In medewerking met DankBank is, aangaande het verzoek om een onderzoek te starten, onderzoek gepleegd naar de beveiliging van de door DankBank gemaakte interface en client. Eerst is de code geanalyseerd, daarna zijn de bevindingen van deze analyse besproken binnen het onderzoeksteam en zijn de problemen, die door het onderzoeksteam ter verbetering vatbaar werden bevonden, opgeschreven en aan de hand daarvan zijn er een aantal adviezen geformuleerd. Uiteindelijk zullen al deze punten de conclusie vormen.

Problemen

Probleem 1:

Binnen de code (zie DankBankGui) is er weinig onderscheid tussen bestanden en staan verschillende bestanden door elkaar, dit maakt het moeilijk leesbaar en veel zoekwerk voor anderen om de code te begrijpen en deze dusdanig te kunnen bewerken of te beoordelen. Hierbij zou categoriseren dit al verhelpen. Bestanden van dezelfde soort in dezelfde map zetten helpt om de overzichtelijkheid te verbeteren.

Ons advies voor probleem 1 is, zorg dat je code leesbaar is. Dit zodat andere het kunnen begrijpen om het verbeteren, uitbreiden of beoordelen. Je kan dit aanpakken door folders aan te maken. In Visual Studio is daar de mogelijkheid voor.

Probleem 2:

De code is normaal geschreven zonder commentaar, voor de leesbaarheid van de code zou het aan te raden zijn om deze te voorzien van commentaar. Op een stuk code of bij enkele regels er extra uitleg, indien nodig, bij schrijven kan dit al helpen. Ook consistent wezen bij het gebruik haakjes zou de code beter leesbaar maken.

Ons advies voor probleem 2 is, plaats commentaar achter een methode. We weten wat het programma doet in grote lijnen, maar bij de klassen moesten we meer tijd nemen om goed te begrijpen wat de klasse doet en waarvoor het gebruikt wordt.

Probleem 3:

Op het moment staan er meer libraries dan nodig zijn. Dit zorgt ervoor dat tijdens het compileren libraries worden aangeroepen die niet gebruikt worden. Deze kunnen weggelaten worden om de werking van het programma te bevorderen. Een voordeel dat hieruit te behalen valt, is dat het programma sneller start.

Probleem 4:

Binnen het bestand Application.cs staat een klasse die niks bevat, deze is totaal onnodig en kan weggelaten worden, aangezien deze nergens aan wordt geroepen. Deze weghalen uit de klasse is het verstandigst. Er was ook een variabele genaamd christian in het programma, om consistent te blijven zou je die een andere naam kunnen geven zoals klant, persoon of pashouder.

Probleem 3 en 4 zijn allebei veroorzaakt door slordigheidsfoutjes. Als advies geven we dan, op het moment dan de code klaar is, om alles na te gaan en ervoor te zorgen dat alle onnodige libraries eruit worden gehaald. Ga alle variabelen langs en zorg ervoor dat er duidelijke benamingen gehandhaafd worden.

Probleem 5:

De benamingen die zijn gekozen voor de class bestanden zijn origineel maar ze geven niet weer wat ze doen ter betrekking tot de code. De bestanden genaamd Kekkeronipizza.cs zou je GUI.cs of Main.cs kunnen noemen. Dat zorgt ervoor dat degene die je code leest weet wat de class doet.

Het is min of meer een vervolg op advies 5 want de classes zouden een andere benaming kunnen krijgen wat de code duidelijker maakt.

Conclusie

Over het algemeen is de beveiliging redelijk, de code is alleen niet geweldig leesbaar en zou eerst aangepast moeten worden op consistentie en dergelijken. Voor de volgende keer zou het overzichtelijker en netter zijn als de bestanden op een gestructureerde manier worden onderverdeeld in mapjes, zodat de reviewer of programmeur het makkelijker snapt zodat hij of zij het uit kan breiden of verbeteren.

Literatuurlijst