Szeged, 2019.07.26.

|  |  |
| --- | --- |
| 1. | 1 - 40 óra |
| 2019. 03. 26. – 2019. 04. 04. | |
| A szakmai gyakorlatomat a software Kft.-nél végeztem, ahova .NET fejlesztői pozícióban munkálkodtam. A felvételi teszt megírását követően az első pár napom a munkakörnyezetem megismerésével történt, melynek során betekintést nyertem a csapatom projektjeibe, megmutatták mit hol tudok elvégezni, kit kell keresni stb. Rögtön kaptam is egy betanuló projektet, melynek célja az volt, hogy megismertesse velem azokat a technológiákat, amit nap mint nap használok már azóta is a munkám során. A feladat egy időjárás előre jelző applikáció elkészítése volt .NET WPF-ben, mely egy REST hívással kéri le az időjárás adatokat az OpenWeatherMap Weather API-jától. Az adathalmazt JSON formátumban kaptam meg, amit deszerializálnom kellet, hogy C# objektumokkal tudjak dolgozni. A feladat továbbá az is volt, hogy az MVVM tervezési mintát használva csináljam meg az applikációt, valamint Unit teszteket is kellett írnom az elkészült függvényeimhez. Ezt az úgynevezett három „A” segítségével kellet tagolnom: arrange, act, assert. Tehát először beállítom a mock adatokat olyanokra, amilyenek az éppen ott levő teszthez kellenek majd példányosítom az adott osztályt. Ezután meghívom az adott metódust. Ezen mérföldkő során ismerkedtem meg a Dependency Injection-nel, melynek segítségével jobban megvalósul a modularitás, kevésbé lesznek egymástól függők az alkalmazás különálló komponensei. Ezen felül itt találkoztam először a factory tervezési mintával, melynek segítségével jobban megvalósul az egységbezárás, a példányosítás felelőssége egy másik objektumra hárul. | |
| 2. | 41 - 80 óra |
| 2019. 04. 04. – 2019. 04. 18. | |
| A harmadik mérföldkő közben ismerkedtem meg a CRUD modellel. A lényege az volt, hogy olyan service-t csináljak, ami akkor is működik, ha például SQLite helyett MSSQL-re váltanék. A legnagyobb nehézséget itt az jelentett, hogy meg kellett ismernem az MSSQL környezetét, hogyan is néznek ki itt a lekérdezések, valamint eligazodni, hogy mi mit jelent a C#-beli conneciton stringben. Ebben a mérföldkőben meg kellett jeleníteni a korábban elmentett időjárást, valamint választhatóvá tenni a felhasználó számára, hogy SQLite vagy MSSQL adatbázist szeretne használni. A feladat implicit specifikációja az is volt, hogy egy bejelentkezés form-ot is megvalósítsak, mely segítségével meg tudja adni a felhasználó az MSSQL adatbázishoz való hozzáféréshez szükséges adatait. Ehhez csak a tesztjeinket kell lefuttatni, amire a ráfordított idő lényegesen kevesebb mint az adott környezetben lebuild-elni, futtatni, a lépéseket megcsinálni, hogy eljussunk az adott teszt esethez újból meg újból. Eleinte nagyon furcsa volt, mivel azt sem tudtam, hogy hogyan álljak neki az egészhez. Eddigi megszokásom az volt, hogy mindig kipróbáltam az adott funkció működését az applikáció elindításával, ami lényegesen sok időt vett igénybe. Meglehetősen nehezen ment ez a rész nekem, viszont a végére én is kezdtem látni ennek az előnyeit, nem kellett tényleg azzal foglalkoznom, hogy mindig elindítsam az adott applikációt, elegendő volt csak megfelelően definiálni a tesztet. | |
| 3. | 81 - 120 óra |
| 2019. 04. 18. – 2019. 05. 09. | |
| A harmadik mérföldkő befejeztével végeztem a betanuló projekttel, így elkezdhettem dolgozni az éles projektünkön. Mielőtt elkezdtem volna foglalkozni az éles projekttel a csapatom kérte, hogy reflektáljak a tanultakra, mi az, aminek láttam a hasznát, mi az, aminek nem és természetesen kellett egy bemutatót tartanom, hogy hogyan is működik az alkalmazás. Elmondták, hogy mi a véleményük, mit kellett volna másképp csinálnom, mi az, amit jól csináltam és milyen területeken kell még fejlődjek. A kritikát vettem és ezzel a lendülettel bemutatták a projektet, amin dolgoznak. A csapatom céges addin-okat fejleszt, melyek többnyire .NET környezetben íródtak. A gerrit-en az a munkamenet, hogy push-olom a változtatásaimat, az megjelenik az én változtatásaim között. Rá kell rakjam munkatársaimat, mint review-ereket, akik átnézik a kódot, hogy mi az, amit rossznak találtak és javítsak ki. Az első commitom egyértelműen nem volt megfelelő, így egy párszor ki kellett javítanom azt. Elsősorban nem volt megfelelő a commit message. A SCRUM módszertana alapján sprintekben dolgozunk, melyek mindegyike két hétig tart. Minden sprint végén nekem is be kellett számolni arról, hogy mit csináltam. Ezek után megtervezzük, hogy mit fogunk csinálni a következő sprintben, milyen JIRA ticket-eket húzzunk be, ki mit fog csinálni. | |
| 4. | 121 - 160 óra |
| 2019. 05. 09. – 2019. 06. 06. | |
| Még pár bug után elkezdtem foglalkozni komolyabb feladatokkal is. Az első nagyobb JIRA ticketem a projektünk legnagyobb komponensének a refactorolása volt, melyet szét kellett szednem kisebb komponensekre. Erre azért volt szükség, mivel minden alkalommal, amikor arra a komponensre került a fejlesztés legalább 2 órával tovább tartott a fejlesztés, mivel annyira átláthatatlan volt az egész. Azért volt ez jó, mivel nagyban megismertem, hogy hogyan is működik az egész kód és ezt közösen csináltam az egyik munkatásammal, hogy ne kelljen egyedül szenvednem és felelősséget vállalnom, ha véletlenül nem sikerül úgy ahogy. Ez az a komponens, amire a kódunk 80%-a kihat, a view, amit az addin paneljén az adatok megjelenítésére szolgál. A legnagyobb kihívást itt nyílván az jelentette, hogy nem ismertem a kódot és nagyon sok mindent eltörtem, amit nem kellett volna. Szerencsére a tesztelő a csapatunkban mindent rendesen átnézett, így nem kellett nagyon aggódnunk, ha talált valami, akkor megmondta, hogy hol a hiba, nekünk csak azzal kellett foglalkoznunk, hogy megmaradjon az eddigi működés viszont másképpen történjen az elhelyezkedése kódban, ne legyen ennyire olvashatatlan. Egyértelműen nagyon sokszor újra kellett kezdenünk, de végül egy sokkal igényesebb dolog alakult ki belőle. Egy másik feladatom az volt, hogy az általunk előállított gbt kiterjesztésű csatolmányok más fajta névvel legyenek csatolva a levélhez, mint amilyenek voltak. Első ránézésre ez egy egyszerű feladat volt, de a csapatvezetőm kérte, hogy ez is vonjon maga után egy refactorálást, mivel nem volt a csatolmány készítő komponensünk megfelelő állapotban; nem volt eléggé objektumorientált. | |
| 5. | 161 - 200 óra |
| 2019. 06. 06. – 2019. 07. 01. | |
| Ez után a következő feladatom az volt, hogy egy teljesen új funkciót implementáljak az addin-ban. A feladat lényege az volt, hogy e-mail írás során az Outlook jobb oldalon található addin-unkon megjelenő CRM adatok közül, azokra jobb-klikkelve ki lehessen választani egy olyan funkciót, hogy az adott adatrekordot hozzáfűzzük hivatkozásként az éppen írott email-hez. A link nem csak egy egyszerű hivatkozás volt, külön dizájn volt rá. A linkre rákattintva megnyílik a felhasználó böngészőjében a céges oldal, ahol az adott rekord található, így azt jobban meg lehet vizsgálni. Az alapkoncepció az volt, hogy átkonvertáljuk az email törzsét html alapúra, így könnyen be tudjuk rakni magát a UI element-et. Ez azért nem volt jó megoldás, mivel nem tudjuk, hogy az adott felhasználó milyen módon kívánja az email törzsét megadni, emiatt ezt az ötletet el kellett vetnünk. Egy másik ötletünk az volt, hogy generálunk egy képet, amin rajta lesz az adott hivatkozáshoz megjelenítendő cím és majd ezt a képet szúrjuk be és alakítjuk át linkké. Ez a megvalósítás két okból kifojólag sem volt megfelelő. Először azért is, mivel a kép kigenerálása nagyon hosszas ideig tartott és nem várhattuk el a felhasználótól, hogy ennyi ideig várjon.Viszonylag egyszerű fix volt, nem kellett vele sokat foglalkozni. Egyedül az volt nehéz benne, hogy több réteg is volt egymáson, így más színű volt a kijelölés, ezért nem a színekkel kellett foglalkozni egy kicsit. | |
| 6. | 201 - 240 óra |
| 2019. 07. 01. – 2019. 07. 15. | |
| Ezt követően a következő fontos feladatom az volt, hogy egy ún. töltőképernyőt hozzak létre az addin-hoz. Magyarán szólva, addig amíg nem érkeznek meg a szerverről az adatok és az addin még nem áll készen addig ezt a felhasználói felületen jelezzük a felhasználó felé valamilyen formában, például animációval. A feladat megoldása elég hosszúra sikeredett, mivel át kellett strukturálni, formázni az XAML fájlok nagy részét a projektben. Külön nehézséget okozott, hogy az animáció behúzása egy külön bug-ot okozott a felhasználói felületen, ami abban nyílvánult meg, hogy miután véget ért a betöltés, az animált elemek látszólag eltűntek, de ottmaradt a helyükön valami, ami a későbbiekben zavart okozott a hover animációkban. Ezt csak egyféleképpen tudtuk kijavítani: végig kellett iterálni a vizuális fán és kikapcsolni az összes ilyen animált elem animációját. Mivel az addin használata során is szükség van erre a töltőképernyőre, így az nem lett volna ésszerű döntés, hogy minden egyes forrás letöltésekor újra és újra bejárjuk a vizuális fát, ezért egyszerűen nem használtuk az animációt, csak szimplán a szürke alakzatokat hagytuk benne. Ha valaki olyan kezdett volna ezen a feladaton dolgozni, aki már régebb óta dolgozik ezen a projekten és már meg van az a tudása, ami ahhoz kell, hogy ne hasaljon el olyan buktatókon, mint amilyeneken én hasaltam el, akkor egy valószínűleg egyszerű feladat lett volna, amit a megadott időn belül el tudok végezni. | |
| 7. | 241 - 280 óra |
| 2019. 07. 15. – 2019. 07. 22. | |
| A következőkben egy másik általunk fejlesztett addin-ban kellett egy új funkciót leimplementálni. És a mail merge csak annyit csinál, hogy legenerálja az adott mezőnevekkel elkészült dokumentumot annyiszor, ahány ember van és kitölti minden ember megfelelő értékével ezeket a mezőket. Ez például hasznos tud lenni, ha sok embernek akarunk feladni egy levelet ugyanazzal a mintával a postán. Az addin-unk is valami hasonlót csinál, de az csak az általunk készített adathalmazt dolgozza fel. A feladatom az volt, hogy leimplementáljak egy olyan funkciót, amely elérhetővé teszi azt a választási lehetőséget, hogy melyik mező alapján rakja a készült dokumentumok sorrendbe. A legnagyobb kihívást az okozta, hogy hol találjam azt a komponenst, ami a csv adatok betöltéséért felelős, illetve hova rakjam azt a logikát, ami azt dönti el, hogy milyen sorrendbe legyenek megcsinálva a merge-k. A static, ill. dynamic mail merge különböző helyeken volt a kódban. Először nem is tudtam, hogy létezik másfajta mail merge. Emiatt a tesztelés után visszakerült hozzám, hogy megcsináljam mindkettőre egységesen. | |
| 8. | 281 - 320 óra |
| 2019. 07. 22. – 2019. 07. 26. | |
| Végezetül, mivel egyszerre több projektet is magában foglal a solution, amit fejlesztünk, így célszerű volt a közös részeket összevonni. Munkatársam leimplementálta azt a logikát, amivel sokkal egyszerűbben el tudjuk dönteni, hogy melyik terméket használjuk éppen. Az én feladatom az volt, hogy azokat az osztályokat, amiknek már nem volt értelme már, hogy külön-külön osztályokban szerepeljenek, összevonjam egy közös osztályba a közös projectbe így mindegyik alkalmazás ugyanazt az egységes logikát használja. Nem volt nagy eltérés a kódok között, mivel nagyjából ugyanazt csinálták, így meglehetősen jól lehetett vele haladni Elég egyszerű feladat volt, viszont nagyon sok tesztjavítási ellátnivalót vont maga után. Valamint tesztelőnek is nagyon sok idő volt mire mindent megnézett, hogy ugyanúgy működik-e minden, mint ahogyan eddig. Ezek után kijavítottam még elég sok support ticketet. A support ticketek a felhasználóktól jönnek a német kollégáinknak és ha azt hibának vélik, akkor továbbítják nekünk visszajelzésként. Arról számolnak be, hogy éppen melyik funkcióban van egy kis hiba. Általában egy részletes log fájllal csatolva érkeznek és a hiba elég pontosan meg van fogalmazva. A feladatot megnehezítette, hogy nem ismertem még az eventek működését és sokat kellett kutatnom, hogy ezt éles alkalmazásokban hogyan szokták megcsinálni. A megoldásra más, kifejezetten erre célra elkészített eventeket használtam és egy olyan WPF konstrukciót, amellyel event-eket lehet küldeni command-ként a ViewModel-ekre, így valóban megvalósul az MVVM tervezési minta és a funkció is működőképes lesz. | |