Green Fox Academy Korlátolt Felelősségű Társaság
Székhely: 1062 Budapest, Andrássy út 66
Nyilvántartási szám: B/2020/000132
Engedélyszám: E/2020/000006

KÉPZÉSI PROGRAM

a Felnőttképzésről szóló 2013. évi LXXVII. tv. 12.§ (1) bekezdésének megfelelően

1. Képzési program

1.1.	A szakmai képzés megnevezése	Junior vállalati Java backend fejlesztő
1.2.	A programkövetelmény megnevezése	Junior vállalati Java backend fejlesztő
1.3.	Programkövetelmény azonosító száma	06135011
1.4.	A képzés célja	A képzés célja, hogy a képzésben résztvevő képes legyen egy alkalmazás többrétegű backendjét felépíteni, mely más alkalmazások által meghívható API-val rendelkezik. Tudjon olyan nagyvállalati keretrendszereket és technológiákat használni, melyek kielégítik a sokszor csak nagyvállalatoknál jelentkező igényeket. Ilyen szabvány pl. a Java Enterprise Edition és ennek implementációi, vagy ezzel kompetitív Spring Framework/Spring Boot keretrendszer, stb. Ismerjen olyan ORM technológiát, mellyel adatbázishoz képes hozzáférni, és tudjon programozott vagy deklaratív tranzakciókezelést használni. El tudja készíteni az alkalmazás üzleti logika rétegét, használva a vállalati backend technológia különböző lehetőségeit (Dependency Injection, Clean Code). Ezen túl képes legyen az alkalmazáshoz szabványokon alapuló API-t fejleszteni. Az alkalmazás komponenseihez unit és integrációs teszteket tudjon írni és futtatni. A fejlesztés során konténerizációs technológiát tudjon használni.

Szignó helye

SZÉCHENYI 2020
MAGYARORSZÁG KORMÁNYA
Európai Unió
Európai Szociális Alap
BEFEKTETÉS A JÖVŐBE

1.5.	A képzés célcsoportja	Minden olyan Junior vállalati Java backend fejlesztő szakképesítést iránt érdeklődő személy, aki megfelel a programba való bekapcsolódás feltételeinek a 3. pont szerint.

2. Képzés során megszerezhető kompetenciák

A képzésben résztvevő:

2.1.	Megfelelően képes használni egy verziókezelő rendszert a hatékony csoportmunkához (Git, Subversion, TFS, stb.)
2.2.	Ismeri a verziókövető rendszer képességeit, parancsait.
2.3.	A lehető leghatékonyabban vesz részt a csoportmunkában a saját munkájának megosztásával.
2.4.	Önállóan használja a verziókövető rendszert saját munkájának publikálására.
2.5.	Az adatokat relációs adatbázisban tárolja és azokhoz hozzáférést biztosít. Ehhez valamilyen ORM technológiát használ (JPA, Hibernate, EclipseLink, stb.), mely automatikusan leképezi az objektumokat táblákra. Ennek használatával alakítja ki a perzisztens réteget.
2.6.	Ismer legalább egy olyan ORM technológiát, mellyel SQL használata nélkül automatikusan tud objektumokat leképezni relációs adatbázisba.
2.7.	Törekszik arra, hogy a megfelelő adatok tárolása és betöltése ORM eszközzel történjen.
2.8.	Önállóan alakít ki adatbázis sémát, majd programból ahhoz kapcsolódik és adatokat ment, módosít, felolvas és töröl valamely ORM eszköz használatával.
2.9.	Objektumok közötti kapcsolatokat is képes perzisztálni.
2.10.	Ismeri, hogy hogy lehet több, egymással kapcsolatban álló entitást adatbázisba menteni. Ismeri a felolvasási stratégiákat (eager, lazy betöltés).
2.11.	Törekszik arra, hogy a kapcsolatok betöltése hatékonyan történjen, csak az legyen betöltve, melyre szükség van.
2.12.	Önállóan dönt a tárolás és betöltés módjáról kapcsolódó entitások esetén.
2.13.	Képes tranzakciókezelést használni (programozott vagy deklaratív módon).
2.14.	Ismeri az adott keretrendszer tranzakciókezelési megoldásait (programozott vagy deklaratív), a kapcsolódó metódusokat és annotációkat.
2.15.	Törekszik arra, hogy hatékony tranzakciókezelést használjon, ne nyújtsa feleslegesen hosszúra a tranzakciókat.
2.16.	Önállóan dönt a tranzakciós határok meghúzásáról.
2.13.	Képes felépíteni egy háromrétegű alkalmazást valamilyen keretrendszer

Szignó helye

	használatával (pl. Spring Framework, Spring Boot, Java EE, valamilyen MicroProfile implementáció, tetszőleges microservice keretrendszer, stb.)
2.14.	Ismer egy háromrétegű alkalmazások felépítésére használható keretrendszert, annak Dependency Injection képességeit.
2.15.	Törekszik arra, hogy az alkalmazásait három réteggel építse fel (prezentációs, üzleti logika, perzisztens). Erre valamilyen keretrendszert használ, ami támogatja a rétegek megvalósítását, azok összekapcsolását.
2.16.	Önállóan épít össze egy háromrétegű alkalmazást, és a komponenseket a megfelelő rétegekben helyezi el.
2.17.	Képes felépíteni egy tisztán objektumorientált, clean code szabályokat alkalmazó üzleti logika réteget. Ezt megfelelően leválasztja a prezentációs és perzisztens rétegtől.
2.18.	Ismeri azokat a technikákat, hogy hogy tud üzleti entitásokat és folyamatokat modellezni és implementálni.
2.19.	Törekszik arra, hogy egy könnyen olvasható és karbantartható, a többi rétegtől független üzleti réteget alakítson ki.
2.20.	Önállóan alakít ki olyan üzleti logika réteget, mely tartalmazza az üzleti elvárások megvalósítását, az üzleti entitásokat és folyamatokat.
2.21.	Klasszikus CRUD műveleteket megvalósító API végpontokat fejleszt (pl. REST webszolgáltatások Spring Framework Controllerek, vagy JAX-RS implementáció használatával, stb.). Ezzel felépíti a prezentációs réteget.
2.22.	Ismer legalább egy vállalati backend technológiát (Spring Controller, JAX-RS, stb.), és hogy hogyan kell vele API-t fejleszteni. Tisztában van az API tervezéssel, van megoldása CRUD műveletek elkészítésére, hibakezelésre, validációra.
2.23.	Törekszik a szabványoknak megfelelő API kialakítására (pl. OpenAPI szabvány, stb). Olyan API-t alakít ki, melyet felületi technológia (pl. JavaScript keretrendszer, mobil vagy vastag kliens) tud használni..
2.24.	Önállóan alakítja ki az alkalmazás API felületét. Dönt a hibakezelés és validáció megvalósításáról.
2.17.	Alkalmazásokat és az adatbázist konténerizált környezetben futtat (pl. Docker).
2.18.	Ismer egy konténerizációs technológiát és hogy hogyan kell konténereket létrehozni és kezelni. Ismeri annak módját, hogy hogyan csomagolja az alkalmazást valamilyen konténerbe.
2.19.	Törekszik arra, hogy az alkalmazásait úgy írja meg, hogy azok konténerizált környezetben is futtathatóak legyenek.
2.20.	Önállóan olyan tervezési döntéseket hoz, hogy az alkalmazás konténerizált környezetben is futtatható legyen.
2.21.	Inicializációs eszközt használ az adatbázis sémájának létrehozására (pl. Flyway, Liquibase, stb.).
2.22.	Ismer egy séma inicializációs eszközt, mely létrehozza az adatbázisban a szükséges

Szignó helye

	objektumokat (táblákat, view-kat, tárolt eljárásokat, stb.).
2.23.	Szem előtt tartja, hogy az alkalmazás maga is képes legyen létrehozni és migrálni az adatbázis sémát.
2.24.	Önállóan adatbázis sémát inicializáló keretrendszert használ.
2.25.	Az alkalmazáshoz, melyet fejleszt, unit és integrációs teszteket ír és futtat valamely keretrendszer használatával (JUnit, TestNG, stb.).
2.26.	Ismer legalább egy unit és integrációs teszt eszközt (JUnit, TestNG, stb.), ismeri azok képességeit, funkcióit.
2.27.	Elkötelezett abban, hogy tesztekkel lefedje az általa készített kódot a használt teszteszköz funkcióinak kihasználásával.
2.28.	Önállóan teszt eszközt választ és teszteseteket ír, melyekkel a funkcionális helyesség biztosítható és automatizáltan újra tesztelhető.
2.29.	A teszteseteket képes különböző környezetekben futtatni (akár fejlesztőeszközben, mint IntelliJ IDEA, Eclipse, NetBeans, stb., akár build eszköz segítségével, mint Maven, Gradle, stb.)
2.30.	Ismeri azokat a lehetőségeket, hogy hogyan lehet futtatni a unit és integrációs teszteket.
2.31.	Törekszik arra, hogy a teszteseteket a fejlesztés közben a fejlesztőeszközből is futtassa, valamint build eszközből, akár parancssor használatával.
2.32.	Önállóan futtat teszt eseteket a fejlesztőeszközből és a parancssorból.
2.33.	A tesztesetekben képes olvasható és hatékony ellenőrzéseket (assert) implementálni (pl. JUnit, TestNG, Hamcrest, AssertJ, stb).
2.34.	Ismert egy eszközt, mellyel könnyen karbantartható assert kifejezéseket tud írni.
2.35.	Törekszik, hogy röviden és tömören fogalmazza meg az ellenőrzési kifejezéseket (assert) valamilyen erre specializált keretrendszer használatával.
2.36.	Önállóan ír assert kifejezéseket, melyek olvashatóbbá teszik az ellenőrzéseket a tesztesetekben.
2.37.	Unit tesztelésnél a kapcsolódó komponenseket mockolja erre használatos keretrendszerrel (pl. Mockito, PowerMock, stb.)
2.38.	Ismer egy mock keretrendszert, mellyel a kapcsolódó komponenseket tudja kicserélni tesztelésre előkészített párjukkal (test double).
2.39.	Törekszik arra, hogy a unit tesztek esetén semmilyen valós másik komponenst ne használjon, hiszen az már integrációs teszt. Erre valamilyen keretrendszert használ.
2.40.	Önállóan helyettesíti a kapcsolódó komponenseket azok tesztelésre előkészített párjukkal.
2.41.	Képes rétegenként külön teszteseteket implementálni valamilyen teszt keretrendszer használatával (JUnit, TestNG, Spring Test, Arquillian, stb.)
2.42.	Ismer egy eszközt, melynek segítségével kevesebb kóddal képes rétegenkénti tesztelést felépíteni.

Szignó helye

2.43.	Törekszik arra, hogy az alkalmazását rétegenként tesztelje. A külön letesztelt rétegek összeépítése után már kisebb a hibalehetőség.
2.44.	Önállóan ír teszteseteket úgy, hogy csak egy adott réteget tesztel vele.

3. A programba való bekapcsolódás feltételei

3.1.	Iskolai előképzettség	Középfokú végzettség
3.2.	Szakmai előképzettség	a 06134005 Junior Java backend fejlesztő szakképesítésben meghatározott szakmai kompetenciák megléte a végzettségről szóló tanúsítvánnyal, vagy ennek hiányában, a képző által szervezett előzetes szintfelmérő vizsga sikeres teljesítésével bizonyítottan.
3.3.	Egészségügyi alkalmassági követelmény	Nem szükséges
3.4.	Szakmai gyakorlat területe és időtartama	-

4. A programban való részvétel feltételei

4.1.	Részvétel követésének módja	Haladási napló, és kapcsolódó, a részvételt igazoló dokumentumok
4.3.	Megengedett hiányzás	Maximum 64 óra hiányzás.
4.3	Egyéb feltételek	-----

5. Tervezett képzési idő

5.4.	Összes óraszám	320

6. Tananyagegységek

	A tananyagegység megnevezése
6.1.	Junior vállalati Java backend fejlesztő

6.1. Tananyagegység

6.1.1.	Tananyagegység (témakör) megnevezése	Junior vállalati Java backend fejlesztő
6.1.2.	Tananyagegység célja	A tananyagegység célja, hogy a képzésben résztvevő képes legyen egy alkalmazás többrétegű backendjét felépíteni, mely más

Szigno helye

		alkalmazások által meghívható API-val rendelkezik. Tudjon olyan nagyvállalati keretrendszereket és technológiákat használni, melyek kielégítik a sokszor csak nagyvállalatoknál jelentkező igényeket. Ilyen szabvány pl. a Java Enterprise Edition és ennek implementációi, vagy ezzel kompetitív Spring Framework/Spring Boot keretrendszer, stb. Ismerjen olyan ORM technológiát, mellyel adatbázishoz képes hozzáférni, és tudjon programozott vagy deklaratív tranzakciókezelést használni. El tudja készíteni az alkalmazás üzleti logika rétegét, használva a vállalati backend technológia különböző lehetőségeit (Dependency Injection, Clean Code). Ezen túl képes legyen az alkalmazáshoz szabványokon alapuló API-t fejleszteni. Az alkalmazás komponenseihez unit és integrációs teszteket tudjon írni és futtatni. A fejlesztés során konténerizációs technológiát tudjon használni.
6.1.3.	Tartalma	Verziókövető rendszer képességei, parancsai. Verziókezelő rendszert használata a hatékony csoportmunkához (Git, Subversion, TFS, stb.). Az adatok tárolása relációs adatbázisban és azokhoz hozzáférés biztosítása. Ehhez valamilyen ORM technológia használata (JPA, Hibernate, EclipseLink, stb.), mely automatikusan leképezi az objektumokat táblákra. Ennek használatával a perzisztens réteg kialakítása. Legalább egy olyan ORM technológia ismerete, mellyel SQL használata nélkül automatikusan lehet objektumokat leképezni relációs adatbázisba. Adatbázis sémák kialakítása és programból ahhoz való kapcsolódás, valamint adatok mentése, módosítása, felolvasása, törlése valamely ORM eszköz használatával. Objektumok közötti kapcsolatok perzisztációja. Több, egymással kapcsolatban álló entitás mentése adatbázisba. Felolvasási stratégiák (eager, lazy betöltés). Tárolás és betöltés módjai kapcsolódó entitások esetén. Tranzakciókezelés használata (programozott vagy deklaratív módon). Egy adott keretrendszer tranzakciókezelési megoldásai (programozott vagy deklaratív), a kapcsolódó metódusok és annotációk. Tranzakciós határok, hatékony tranzakciókezelés. Egy háromrétegű alkalmazás felépítése valamilyen keretrendszer használatával (pl. Spring Framework, Spring Boot, Java EE, valamilyen MicroProfile implementáció, tetszőleges microservice keretrendszer, stb.). Egy háromrétegű alkalmazások felépítésére használható keretrendszer, valamint annak Dependency Injection képességei. Egy háromrétegű alkalmazás összeépítése, és a komponensek elhelyezése a megfelelő rétegekben. Üzleti entitások és folyamatok modellezése és implementálása. Tisztán objektumorientált, clean code szabályokat alkalmazó üzleti logika réteg felépítése, leválasztása a prezentációs és perzisztens

Szignó helye

		rétegtől. Olyan üzleti logikai réteg kialakítása, mely tartalmazza az üzleti elvárások megvalósítását, az üzleti entitásokat és folyamatokat. Egy vállalati backend technológia ismerete (Spring Controller, JAX-RS, stb.): klasszikus CRUD műveleteket végrehajtó API fejlesztése, hibakezelés, validáció, prezentációs réteg felépítése. Alkalmazások API felületének szabványos kialakítása (pl. OpenAPI szabvány, stb). Felületi technológia (pl. JavaScript keretrendszer, mobil vagy vastag kliens) által használható API kialakítása. Alkalmazások és adatbázis futtatása konténerizált környezetben: egy konténerizációs technológia ismerete, konténerek létrehozása és kezelése, alkalmazás konténerbe csomagolása, konténerizált környezetben is futtatható alkalmazások tervezése és írása. Inicializációs eszköz használata az adatbázis sémájának létrehozására: egy séma inicializációs eszköz ismerete, mely létrehozza az adatbázisban a szükséges objektumokat (táblákat, view-kat, tárolt eljárásokat, stb.). Olyan technológia használata, mely az adatbázis sémát automatizáltan létrehozza és migrálja. Adatbázis sémát inicializáló keretrendszer használata. Unit és integrációs tesztek írása és futtatása: legalább egy unit és integrációs teszt eszköz ismerete (JUnit, TestNG, stb.), azok képességei, funkciói. Választás a teszteszközök között, tesztesetek írása, melyekkel a funkcionális helyesség biztosítható és automatizáltan újra tesztelhető. A teszteseteket futtatása különböző környezetekben (akár fejlesztőeszközben, mint IntelliJ IDEA, Eclipse, NetBeans, stb., akár build eszköz segítségével, mint Maven, Gradle, stb.). Unit és integrációs tesztek. Tesztesetek futtatása a fejlesztőeszközből és a parancssorból. A tesztesetekben olvasható és hatékony ellenőrzések (assert) implementálása (pl. JUnit, TestNG, Hamcrest, AssertJ, stb). Könnyen karbantartható assert kifejezések írása, melyek olvashatóbbá teszik az ellenőrzéseket a tesztesetekben. Unit tesztelésnél a kapcsolódó komponensek mockolása erre használatos keretrendszerrel (pl. Mockito, PowerMock, stb.). Rétegenként külön tesztesetek implementálása valamilyen teszt keretrendszer használatával (JUnit, TestNG, Spring Test, Arquillian, stb.). Rétegenkénti tesztelésre alkalmas tesztesetek írása.
6.1.4.	Terjedelme	320 óra
6.1.5.	Képzési módszerek	- **Távolléti csoportos és egyéni témafeldolgozás és feladat megoldás konzultációs lehetőséggel.** - **Távolléti önálló és csoportos projektmunka konzultációs lehetőséggel.**

Szignó helye

		- **Online tananyagok önálló feldolgozása, feladat megoldás (házifeladat).** - **Mentori konzultáció.**
6.1.6.	Munkaformák	- Távolléti csoportos (mikro-csoportos, kiscsoportos, projektcsoportos) és egyéni munkaforma.
6.1.7.	Beszámítható óraszám (kontaktórától eltérő munkaforma alkalmazása esetén)	320 óra
6.1.8.	A tananyagegység elvégzéséről szóló igazolás kiadásának feltételei	A képzés egy tananyagegységből áll. A 8. pontban rögzítettek alapján.

7. Csoportlétszám

7.1.	Maximális csoportlétszám (fő)	40

8. A képzésben résztvevő teljesítményét értékelő rendszer leírása

A képzés megkezdése előtt a képzésben résztvevő/képzésre jelentkező kérheti előzetesen megszerzett tudásának mérését, amelyet a felnőttképző a képzés során alkalmazott mérési, értékelési módszerek alkalmazásával végez.

Az önálló tanulás eredményeinek ellenőrzése: A tananyagban és/vagy a tanulási útmutatóban meghatározott tanulási célok alapján, a tananyagba épített önellenőrző feladatok és gyakorlati feladatok sikeres teljesítésének oktatói/mentori ellenőrzése (történhet szintfelmérő teszt, részvizsga és záróvizsga keretében).

A képzés során a résztvevők írásbeli és/vagy szóbeli részvizsgán és egy záróvizsgán adnak számot tudásukról. A részvizsga és a záróvizsga online felületen történik.
A záróvizsgára bocsátás feltétele a részvizsga sikeres teljesítése.
A részvizsga és a záróvizsga projektfeladat (gyakorlati feladat), amely kiegészülhet teszt feladatokkal.

Valamennyi vizsga akkor sikeres, amennyiben a résztvevő a megszerezhető összes pontszám legalább 51%-át elérte.

Szignó helye

A vizsgák minősítése:

Minősítés	Minősítési szempontok
Megfelelt	• A vizsgafeladatot önállóan, a megadott időtartam alatt végrehajtja és eléri a megszerezhető összes pontszám legalább 51%-át.
Nem felelt meg	• A vizsgafeladatot nem tudja önállóan, a megadott időtartam alatt végrehajtani, vagy nem éri el a megszerezhető összes pontszám legalább 51%-át.

9. A képzés zárása

9.1.	A képzés elvégzéséről szóló tanúsítvány kiadásának feltételei	- a záróvizsgán *"Megfelelt"* minősítés elérése a 8. pontban meghatározottak szerint - a résztvevő hiányzása nem haladta meg a 4.3. pontban meghatározott megengedett hiányzást - a felnőttképzési szerződésben vállalt kötelezettségek teljesítése

10. A képzési program végrehajtásához szükséges feltételek

10.1.	Személyi feltételek	Olyan oktatót alkalmazunk, aki rendelkezik felsőfokú informatikai végzettséggel vagy Vállalati Backend technológiákban szerzett legalább 3 éves szakmai tapasztalattal. A képzés során az oktatók munkáját mentorok segíthetik.
10.1.1.	Személyi feltételek biztosításának módja	A személyi feltételeket munkaviszony vagy egyéb foglalkoztatásra irányuló jogviszony alapján (megbízási szerződés, vállalkozási szerződés) biztosítjuk.
10.2.	Tárgyi feltételek	Online képzési alkalmak megvalósításra alkalmas rendszer; Verziókezelő rendszer; A résztvevők saját számítógéppel előre telepített böngészővel, feltelepített Java SDK-val, valamint fejlesztőeszközzel és SQL adatbázissal rendelkeznek.

Szignó helye

10.2.1.	Tárgyi feltételek biztosításának módja	A tárgyi feltételekkel való rendelkezést részben saját tulajdonú eszközökkel, részben bérleti szerződések kötésével biztosítjuk. A 10.2. pontban a résztvevő saját számítógépével kapcsolatban meghatározott feltételeket a résztvevő biztosítja.
10.3.	Egyéb speciális feltételek	-
10.3.1.	Egyéb speciális feltételek biztosításának módja	-

Minősítés helye: Budapest

Minősítés időpontja: 2021.06.11.

A szakértő neve: Bertalan Tamás

A szakértő nyt. száma: FSZ/2020/000073

.....

felnőttképző képviselőjének aláírása

.....

felnőttképzési szakértő aláírása

.....

felnőttképző szakmai vezetőjének aláírása

GREEN FOX ACADEMY KFT.
1062 Budapest, Andrássy út 66.
Adószám: 25374926-2-42
Cg.: 01-09-271502

GREEN FOX ACADEMY

Szignó helye

KÉPZÉSI ÜTEMEZÉS[1]

Junior vállalati Java backend fejlesztő

(06135011)

Képzési program melléklet - A Felnőttképzésről szóló 2013.évi LXXVII.tv.12§(1) bekezdésének megfelelően

A képzés hétköznapokon a következő tevékenységekkel zajlik:

9.00 - 18.00 között napi 8 órában (1 óra ebédszünettel):

- Távolléti csoportos és egyéni témafeldolgozás és feladat megoldás konzultációs lehetőséggel.
- Távolléti önálló és csoportos projektmunka konzultációs lehetőséggel.
- Online tananyagok önálló feldolgozása, feladat megoldás (házifeladat).

Foglalkozások száma és óraszáma	Foglalkozások megnevezése	Témakörök megnevezése
1.-20. Foglalkozás, 160 óra	- Távolléti csoportos és egyéni témafeldolgozás és feladat megoldás konzultációs lehetőséggel. - Távolléti önálló és csoportos projektmunka konzultációs lehetőséggel. - Online tananyagok önálló feldolgozása, feladat megoldás (házifeladat).	Projekt és csapatmunka alapjai
		Backend enpointok készítése
		Többrétegű backend alkalmazások fejlesztése
		Adatbázis elérés ORM rendszer segítségével
RÉSZVIZSGA		
Pótvizsga 1.		
Pótvizsga 2.		

[1] Tartalmazza a megtanítandó és elsajátítandó témaköröket, az ehhez szükséges foglalkozások megnevezését, számát

Szignó helye

21.- 35. Foglalkozás, 120 óra	- Távolléti csoportos és egyéni témafeldolgozás és feladat megoldás konzultációs lehetőséggel. - Távolléti önálló és csoportos projektmunka konzultációs lehetőséggel. - Online tananyagok önálló feldolgozása, feladat megoldás (házifeladat).	Adatbázis migrációk használata
		Automatizált tesztek készítése
		Konténerizáció
36.- 40. Foglalkozás, 40 óra	**ZÁRÓVIZSGÁRA felkészítés:** Mentori konzultáció, Távolléti csoportos és egyéni témafeldolgozás és feladat megoldás konzultációs lehetőséggel, Online tananyagok önálló feldolgozása, feladat megoldás (házifeladat).	
ZÁRÓVIZSGA Pótvizsga 1. Pótvizsga 2.		

Minősítés helye: Budapest

Minősítés időpontja: 2021.06.11.

A szakértő neve: Bertalan Tamás

A szakértő nyt. száma: FSZ/2020/000073

.....

felnőttképző képviselőjének aláírása

.....

felnőttképzési szakértő aláírása

.....

felnőttképző szakmai vezetőjének aláírása

GREEN FOX ACADEMY KFT.
1062 Budapest, Andrássy út 66.
Adószám: 25374926-2-42
Cg.: 01-09-271502

GREEN FOX ACADEMY

Szignó helye

ELŐZETES SZAKÉRTŐI MINŐSÍTÉS

Szakértői vélemény

a szakképzésről szóló törvény szerinti szakmai képzés képzés programjához

Felnőttképző neve	Green Fox Academy Kft.
Székhelye (címe)	1062 Budapest, Andrássy út 66.
Nyilvántartási szám	B/2020/000132
Engedélyszám	E/2020/000006
Program megnevezése	Junior vállalati Java backend fejlesztő
Programkövetelmény azonosítója	06135011
Tervezett képzési idő	320 óra
Tananyagegységek száma	1

A Green Fox Academy Kft. megbízása alapján – a képzési program előzetes szakértői minősítése érdekében – átvizsgáltam a Junior vállalati Java backend fejlesztő szakképzésről szóló törvény szerinti szakmai képzés képzési programját. A rendelkezésemre bocsátott képzési program szakmai átvizsgálása után az alábbi szakértői véleményt alakítottam ki:

Az előzetes minősítése során megállapítottam, hogy a képzési program tartalma megfelel az Fktv.-nek és a szakképzésről szóló törvény és a szakképzésről szóló törvény végrehajtásáról szóló kormányrendelet követelményeinek, összhangban van a Junior vállalati Java backend fejlesztő szakképesítés megszerzésére irányuló szakmai képzéseket megalapozó programkövetelménnyel és a képzési programban meghatározott tartalommal, feltételekkel. A képzési programban rögzítettek alapján a képzéssel érintett célcsoport számára megszerezhetők a képzési programban megjelölt kompetenciák.

A képzés célja, hogy a képzésben résztvevő képes legyen egy alkalmazás többrétegű backendjét felépíteni, mely más alkalmazások által meghívható API-val rendelkezik. Tudjon olyan nagyvállalati keretrendszereket és technológiákat használni, melyek kielégítik a sokszor csak nagyvállalatoknál jelentkező igényeket. Ilyen szabvány pl. a Java Enterprise Edition és ennek implementációi, vagy ezzel kompetitív Spring Framework/Spring Boot keretrendszer, stb. Ismerjen olyan ORM technológiát, mellyel adatbázishoz képes hozzáférni, és tudjon programozott vagy deklaratív tranzakciókezelést használni. El tudja készíteni az alkalmazás üzleti logika rétegét, használva a vállalati backend technológia különböző lehetőségeit (Dependency Injection, Clean Code). Ezen túl képes legyen az alkalmazáshoz szabványokon alapuló API-t fejleszteni. Az alkalmazás komponenseihez unit és integrációs teszteket tudjon írni és futtatni. A fejlesztés során konténerizációs technológiát tudjon használni.

A képzési program egyértelműen kijelöli a képzés során megszerezhető kompetenciákat, a képzés elvégzése utáni többlet tudás/képesség konkrétan meghatározott.

A képzés a felnőttképzésről szóló 2013. évi LXXVII. törvény 2/A 2. a)§ a bekezdése alapján a szakképzésről szóló törvény szerinti szakmai képzés.

A képzési program tartalmi elemeiben megfelel a felnőttképzési törvényben foglaltaknak, valamennyi – a felnőttképzési törvény 12. §-ában foglalt – tartalmi elemet tartalmazza.

Szignó helye

A képzési program a vonatkozó jogszabályoknak megfelelő módon tartalmazza

- a képzés megnevezését,
- a képzés során megszerezhető kompetenciákat,
- a képzésbe történő bekapcsolódás és részvétel feltételeit,
- a tervezett képzési időt (320 óra),
- a tananyagegységeket és azok tartalmi bemutatását,
- a maximális csoportlétszámot (40 fő),
- a képzésben résztvevő teljesítményét értékelő rendszer leírását,
- a képzési program végrehajtásához szükséges személyi és tárgyi feltételeket és azok biztosításának módját.

A képzési program nemcsak a jogszabályi előírásoknak, de **a koherencia követelményeknek is megfelel**, azaz a képzési programban meghatározott tartalommal, feltételekkel és módon, a programban megjelölt kompetenciák az adott időtartam alatt megszerezhetők. A képzés során megszerezhető kompetenciák, a tananyagegység célja és tartalma valamint a teljesítményértékelési rendszer koherens rendszert képez.

A képzési program megfelel a felnőttképzési törvény által előírt tartalmi és formai követelménynek

Az előzetes szakértői minősítést a felnőttképzésről szóló 2013. évi LXXVII. törvény valamint a 11./2020. kormányrendelet felnőttképzésről szóló törvény végrehajtásáról vonatkozó előírásai alapján készítettem el.

Jelen előzetes szakértői minősítés elválaszthatatlan részét képezi az előzetes minősítést végző felnőttképzési szakértő, a felnőttképzést folytató intézmény képviseletére jogosult személy és a szakmai vezető által aláírt képzési programnak.

Minősítést végző szakértő neve: Bertalan Tamás

Minősítés helye: Budapest

Minősítés ideje: 2021.06.11.

Felnőttképzési szakértői nyilvántartási szám: FSZ/2020/000073

Előzetes minősítést végző szakértő aláírása:

.....

Szignó helye