

CardService telepítési útmutató WebLogic szerveren való használatához

Tartalomjegyzék

CardService telepítési útmutató WebLogic szerveren való használatához	1
Az alkalmazás bemutatása	1
Előzetes követelmények	1
Adatbázis létrehozása	1
Alkalmazás konfiguráció módosítása	3
Alkalmazás build-elése, .war fájl előállítás	4
Telepítés WebLogic szerverre	4
Alkalmazás használata	8

Az alkalmazás bemutatása

A CardService egy Spring Boot-alapú, REST API-n keresztül kommunikáló alkalmazás, mellyel segítségével bankkártyákat kezelhetünk és tárolhatunk Oracle SQL adatbázisban. Az alkalmazás támogatja új kártyák létrehozását, meglévő kártyák adatainak lekérdezését, kártyák validálását (a kártyaadatok érvényességének ellenőrzését), illetve kártyák letiltását (egyirányú műveletként).

Az alkalmazás önállóan is futtatható (telepített Java futtatókörnyezet és Oracle SQL szerver megléte esetén), de ebben az útmutatóban a program Oracle WebLogic szerveren való használatát ismerhetjük meg.

Előzetes követelmények

A CardService telepítéséhez és használatához a következő feltételeknek kell teljesülnie:

- Telepített Java SE 8 környezet
- Telepített Oracle SQL szerver (Tesztelt verzió: 11G Express Edition 11.2.0.2.0 64 bit)
- Telepített Oracle WebLogic szerver (Tesztelt verzió: 12.2.1.3.0)

Adatbázis létrehozása

Az alkalmazás által használt adatbázis felépítéséhez két SQL scriptet kell futtatni, melyek a `\cardservice\sql` mappában találhatóak. A képeken található parancsok az Oracle SQL*Plus alkalmazás használatán keresztül mutatják be az egyes lépéseket.

- Első lépésként az adatbázisrendszeren belül egy olyan felhasználóval kell belépünk, amelynek van jogosultsága új felhasználó létrehozására, táblaterület (tablespace) létrehozására, illetve *SESSION*, *TABLE*, *UNLIMITED TABLESPACE* ÉS *SEQUENCE* jogok

megadására. Például, használhatjuk a **SYSTEM** felhasználót, mely rendelkezik ezekkel a jogosultságokkal.

```
SQL> connect;
Enter user-name: system
Enter password:
Connected.
```

- Futtassuk a **00_create_user.sql** fájlt, mely létrehoz egy **CARD** nevű felhasználót **dbadmin** alapértelmezett jelszóval, és létrehozza a kártyaadatok tárolásához szükséges ideiglenes és végleges táblaterületeket (a fájl útvonala a különböző számítógépeken eltérő).

```
SQL> @C:\Users\Robi\Documents\cardservice\sql\00_create_user.sql
```

Figyelem! A script a felhasználó és táblaterület létrehozása előtt automatikusan törli a CARD nevű felhasználót és a card_tablespace nevű táblaterületet, valamint ezek tartalmát is, amennyiben már léteznek.

- Lépünk át az újonnan létrejött **CARD** nevű felhasználóba (az alapértelmezett jelszó: **dbadmin**).

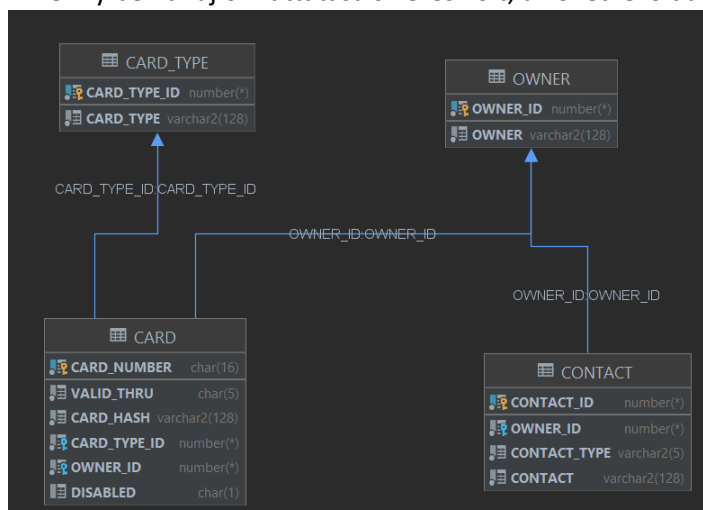
```
SQL> connect;
Enter user-name: card
Enter password:
Connected.
```

- Futtassuk a **01_create_database.sql** fájlt, mely létrehozza a kártyaadatok tárolásához szükséges táblákat, és feltölti a CARD_TYPE táblát a fájlban szereplő kártyatípusokkal.

```
SQL> @C:\Users\Robi\Documents\cardservice\sql\01_create_database.sql
```

Figyelem! A script a táblák létrehozása előtt automatikusan törli a már meglévő azonos nevű táblákat ÉS azok tartalmát is (amennyiben már léteznek).

- Amennyiben a fájlok futtatása sikeres volt, a következő adatbázis-szerkezetet kell kapnunk:



A CARD, CONTACT és OWNER táblákban a frissen létrehozott adatbázis esetén nincs adat. A CARD_TYPE adatbázisban a 01-es SQL fájlban meghatározott kártyatípusok ID-je és neve szerepel.

Alkalmazás konfiguráció módosítása

- Az alkalmazás build-elése előtt be kell állítanunk az adatbázis szerver elérhetőségét a konfigurációban. A `\cardservice\application\cardservice\build\resources\main` mappában nyissuk meg az `application.properties` fájlt (egy egyszerű szövegszerkesztőt is használhatunk).
- A `#JDBC properties` címszó alatt találhatjuk az adatbázis elérhetőségére vonatkozó beállításokat, amennyiben szükséges, változtassuk meg a következő tulajdonságokat:
 - A `spring.datasource.url` tulajdonság az adatbázis szerver elérési útját tartalmazza.
 - A `spring.datasource.username` tulajdonság az adatbázishoz való csatlakozáshoz szükséges felhasználónevet tartalmazza. Ez alapértelmezetten a **CARD** felhasználó.
 - A `spring.datasource.password` tulajdonság a felhasználónévhez tartozó jelszót tartalmazza. Alapértelmezetten **dbadmin** a jelszó.
- **(OPCIONÁLIS)** A `logback-spring.xml` fájlt megnyitva beállíthatjuk a program logolási szintjét, ami meghatározza, hogy milyen üzenetek kerüljenek kiírásra és lejegyzésre a program által. A fájl alján a következő sorokat találhatjuk meg:

```
<!-- LOG "hu.robi.cardservice.*" at DEBUG level to CONSOLE and FILE -->
<logger name="hu.robi.cardservice" level="debug" additivity="false">
  <appender-ref ref="RollingFile"/>
  <appender-ref ref="Console"/>
</logger>
```

Ezen belül a `level=` rész adja meg, hogy milyen részletességgel szeretnénk a konzolablakba és egy szövegfájlba üzeneteket kapni az egyes eseményekről. A választható opciók (a legkevésbé részletestől a legrészletesebbig):

- **level="error"** – Csak végzetes, a program helyes működését megakadályozó hibákról kapunk értesítést.
- **level="warn"** – A végzetes hibák mellett azokról a sikertelen műveletekről is értesítést kapunk, melyeket nem a program hibás működése okoz (pl. hibásan megadott kártyaadatok)
- **level="info"** – Az előbbiek mellett a sikeres műveletekről, illetve (pl. validálás esetén) azok eredményeiről is kapunk értesítést.
- **level="debug"** – Az előbbiek mellett a program belső folyamatairól is kapunk üzeneteket. Használata csak hibakeresés esetén ajánlott.

Alkalmazás build-elése, .war fájl előállítás

Ahhoz, hogy az alkalmazást telepíteni tudjuk egy WebLogic szerverre, a forráskódból elő kell állítani a kész, futtatható alkalmazást .war formátumban. Ehhez a gradle nevű alkalmazást tudjuk használni, mely a forráskóddal együtt mellékelve van a programban. **Fontos, hogy a zökkenőmentes előállítás érdekében a forráskódot egy olyan mappába kell elhelyeznünk, melynek írásához van hozzáférésünk normál felhasználóként is.**

Windows rendszer használata esetén, a kész program előállításának legegyszerűbb módja, ha a `\cardservice\application\cardservice` mappán belül, a `build_war_file.bat` nevű, futtatható állományt használjuk. Csak kattintsunk rá a fájlra, amely létrehozza nekünk a kész alkalmazást.

Amennyiben nem Windows rendszert használunk (vagy csak szeretnénk egyedi beállításokat használni), az alkalmazás előállítását manuálisan is elvégezhetjük. Ehhez nyissunk meg egy parancssort/terminált a korábban említett mappában, majd adjuk ki a következő parancsot:

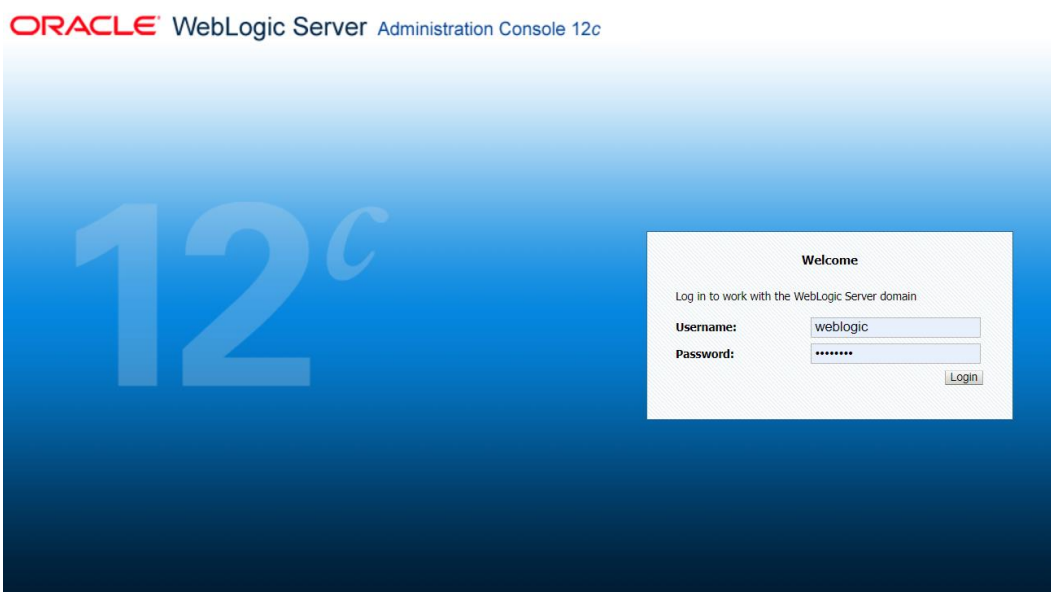
- `gradlew bootWar` (Windows esetén)
- `./gradlew bootWar` (Unix alapú rendszerek, pl. Linux esetén)

Bármelyik módszert válasszuk, a folyamat végén egy zöld **BUILD SUCCESSFUL** üzenet jelzi, ha a build folyamat sikerült (ha a kattintható fájlt használtuk, a parancssor ezután automatikusan bezárul). Ebben az esetben, a kész programot a `\build\libs\` mappában érjük el, ahol egy `cardservice-x.y.z.war` nevű fájlt találunk (az x.y.z az aktuális verziót jelöli).

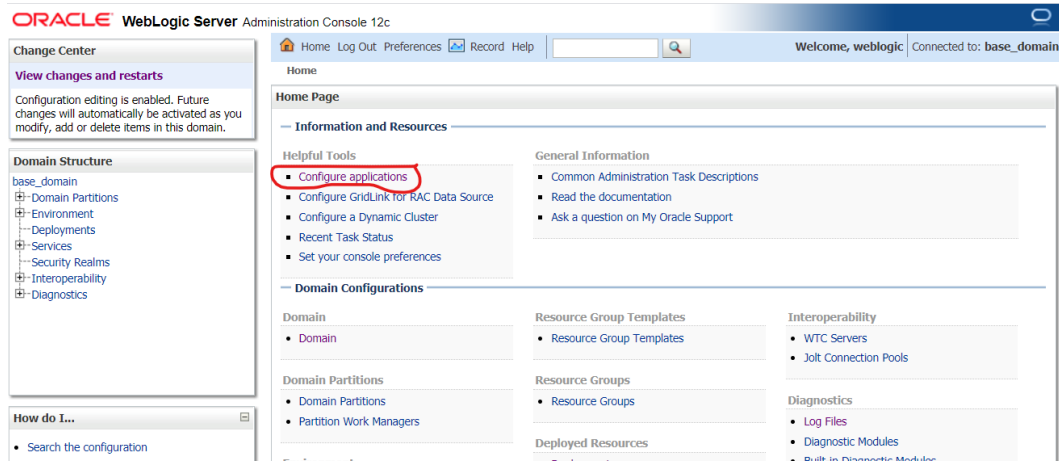
Telepítés WebLogic szerverre

Az alkalmazás telepítéséhez, be kell lépünk a WebLogic szerver konfigurációs felületére. Amennyiben a szerver már fut, egy webböngészőben látogassunk el a <http://<szervercíme>:<szerverport>/console> címre. Például, ha a saját számítógépünkön, az alapértelmezett porton futtatjuk a szervert, a <http://localhost:7001/console> címet használhatjuk.

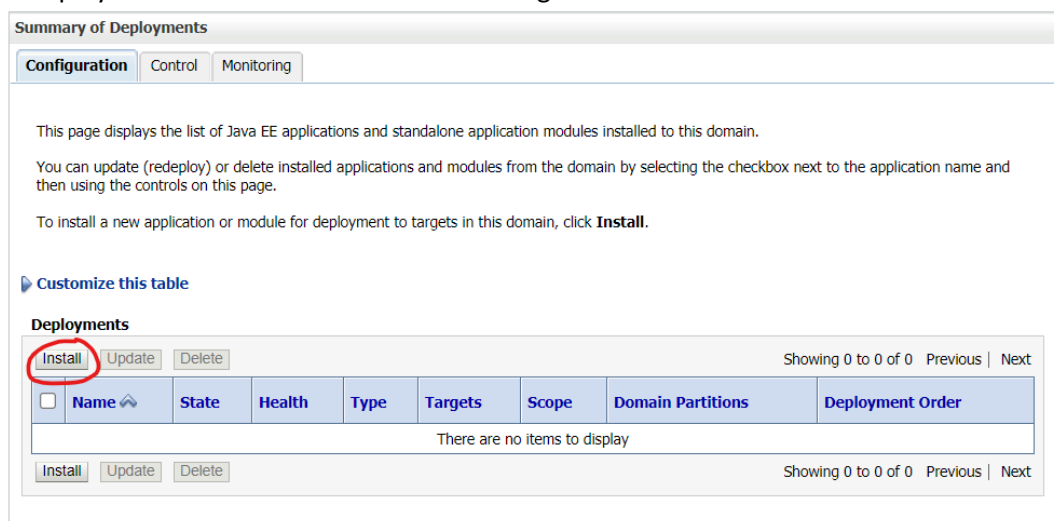
- A megjelenő weboldalon jelentkezünk be a szerverhez tartozó felhasználónévvel és jelszóval (alapértelmezett felhasználónév **weblogic**, alapértelmezett jelszó **w3blogic**).



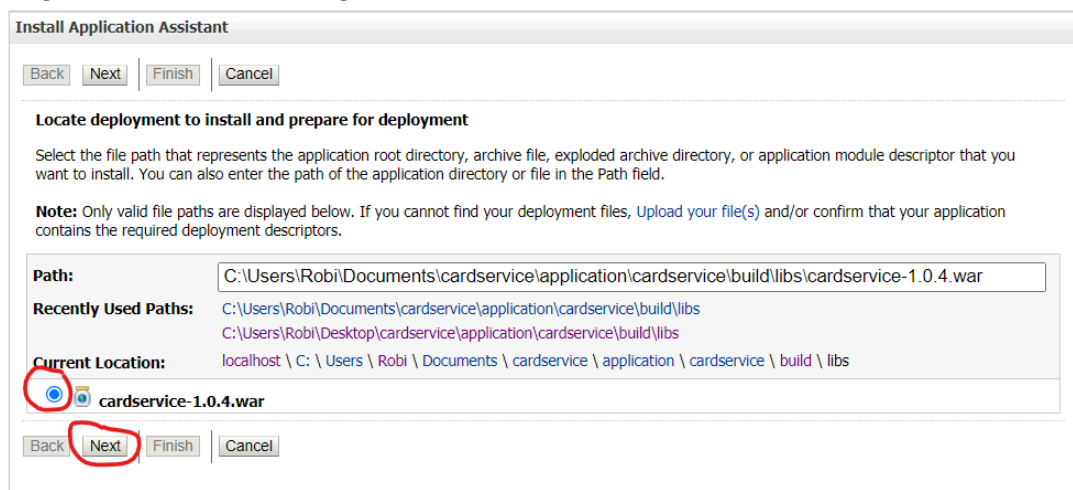
- Ha a belépés sikerült, a megjelenő oldalon kattintsunk a **Configure applications** linkre.



- A Deployments ablakban válasszuk az **Install** gombot.



- A weboldalon található fájlkezelő segítségével navigáljunk el a futtatható fájlt tartalmazó mappába (`\cardservice\application\cardservice\build\libs`), majd **válasszuk ki** a telepíteni kívánt .war fájlt. (a konkrét elérési út és a fájl neve eltérhet a képen láthatótól). Ha ez megvan, kattintsunk a **Next** gombra.



- A következő oldalon válasszuk az **Install this deployment as an application** lehetőséget.

Install Application Assistant

Back Next Finish Cancel

Choose installation type and scope

Select if the deployment should be installed as an application or library. Also decide the scope of this deployment.

The application and its components will be targeted to the same locations. This is the most common usage.

☒ **Install this deployment as an application**

Application libraries are deployments that are available for other deployments to share. Libraries should be available on all of the targets running their referencing applications.

☐ **Install this deployment as a library**

Select a scope in which you want to install the deployment.

Scope: Global ▾

Back Next Finish Cancel

- A következő oldalon **opcionális beállításokat** módosíthatunk. Amennyiben erre nincs szükségünk, **hagyjuk az egyes opciókat az alapértelmezett beállításokon**, majd a **Finish** gombot használva fejezzük be a telepítést. A gomb megnyomása után a következő oldal betöltése tovább tarthat, mivel ekkor történik meg az alkalmazás telepítése és elindítása.

Install Application Assistant

Back Next Finish Cancel

Optional Settings

You can modify these settings or accept the defaults.

* Indicates required fields

General

What do you want to name this deployment?

* **Name:** cardservice-1.0.4

Security

What security model do you want to use with this application?

☒ **DD Only: Use only roles and policies that are defined in the deployment descriptors.**

☐ **Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor.**

☐ **Custom Roles and Policies: Use only roles and policies that are defined in the Administration Console.**

☐ **Advanced: Use a custom model that you have configured on the realm's configuration page.**

Source Accessibility

How should the source files be made accessible?

☒ **Use the defaults defined by the deployment's targets**

Recommended selection.

☐ **Copy this application onto every target for me**

During deployment, the files will be copied automatically to the Managed Servers to which the application is targeted.

☐ **I will make the deployment accessible from the following location**

Location: C:\Users\Robi\Documents\cardservice\application\cardserv

Provide the location from where all targets will access this application's files. This is often a shared directory. You must ensure the application files exist in this location and that each target can reach the location.

Plan Source Accessibility

How should the plan source files be made accessible?

☒ **Use the same accessibility as the application**

Recommended selection.

☐ **Copy this plan onto every target for me**


During deployment, the plan files will be copied automatically to the Managed Servers to which the application is targeted.

☐ **Do not copy this plan to targets**

You must ensure the plan files exist in the shared location and that each target can reach the location.

Back Next **Finish** Cancel

- Ha mindent jól csináltunk, visszajutunk a futó alkalmazások listájához, ahol megjelenik az általunk telepített program.

Install Update Delete			Showing 1 to 1 of 1 Previous Next					
<input type="checkbox"/>	Name	State	Health	Type	Targets	Scope	Domain Partitions	Deployment Order
<input type="checkbox"/>	 cardservice-1.0.4	Active	OK	Web Application	AdminServer	Global		100
Install Update Delete			Showing 1 to 1 of 1 Previous Next					

Amennyiben van hozzáférésünk a WebLogic server konzolablakához, azt is láthatjuk, ahogy a program elindul. Ugyanitt fogjuk látni a program futása során megjelenő log üzeneteket is.

```

  _ _ _ _ _
 / _ _ _ _ \
( ( _ _ _ _ )
 ' _ _ _ _ '
  _ _ _ _ _
  :: Spring Boot ::
                (v2.4.4)

2021-04-30 12:17:17 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO h.r.cardserv
ice.ServletInitializer - Starting ServletInitializer using Java 1.8.0_281 on ROBI-PC with PID 6328 (C:\oracleserver\user
_projects\domains\base_domain\servers\AdminServer\tmp\WL_user\cardservice-1.0.4\qnq860\war\WEB-INF\lib\wl_cls_gen.jar
started by Robi in C:\oracleserver\user_projects\domains\base_domain)
2021-04-30 12:17:17 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] DEBUG h.r.cardserv
ice.ServletInitializer - Running with Spring Boot v2.4.4, Spring v5.3.5
2021-04-30 12:17:17 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO h.r.cardserv
ice.ServletInitializer - No active profile set, falling back to default profiles: default
2021-04-30 12:17:19 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO o.s.d.r.c.Re
positoryConfigurationDelegate - Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2021-04-30 12:17:19 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO o.s.d.r.c.Re
positoryConfigurationDelegate - Finished Spring Data repository scanning in 168 ms. Found 3 JPA repository interfaces.
2021-04-30 12:17:20 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO o.s.b.w.s.c.
ServletWebServerApplicationContext - Root WebApplicationContext: initialization completed in 2250 ms
2021-04-30 12:17:21 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO com.zaxxer.h
ikari.HikariDataSource - HikariPool-3 - Starting...
2021-04-30 12:17:21 [[STANDBY] ExecuteThread: '4' for queue: 'weblogic.kernel.Default (self-tuning)'] INFO com.zaxxer.h
ikari.HikariDataSource - HikariPool-3 - Start completed.

```

A program alapértelmezés szerint a WebLogic serverrel megegyező címen és porton, a /cardservice/ kiterjesztéssel érhető el.

Például, amennyiben a saját számítógépünkön, az alapértelmezett porton fut a WebLogic serverünk, a következő címet használhatjuk az alkalmazás elérésére: <http://localhost:7001/cardservice/>

Az alkalmazás használatához a következő fejezetben találhatunk segítséget.

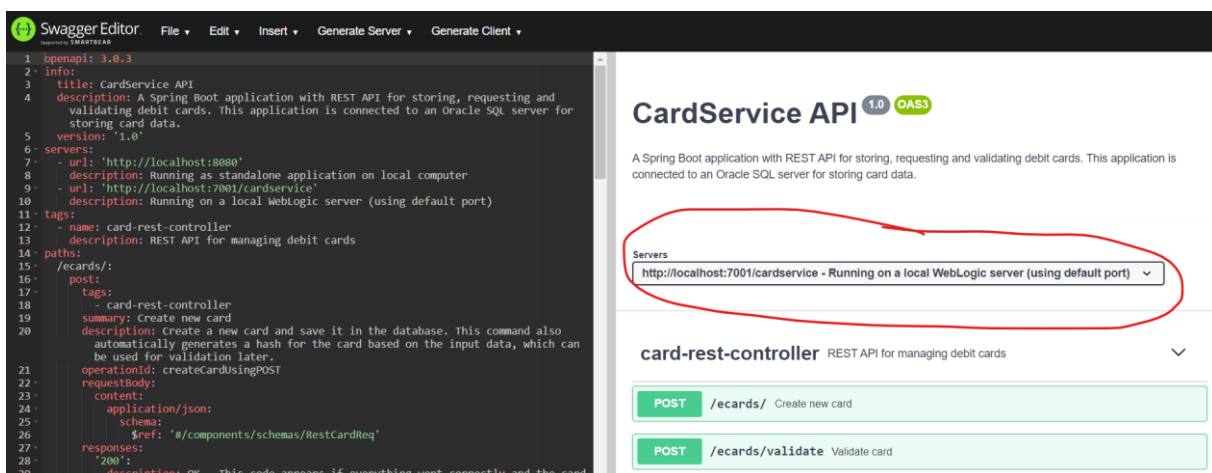
Alkalmazás használata

Amennyiben a telepítés sikeres volt, az alkalmazást elkezdhetjük használni. Az alkalmazás funkcióinak eléréséhez egy REST műveleteket támogató klienst kell használnunk (pl. Postman).

A használathoz segítséget a Swagger dokumentációban találhatunk, amely egy beépített REST klienst is tartalmaz, így kipróbálhatjuk az egyes szolgáltatásokat (fontos tudnunk, hogy ezek a műveletek is módosíthatják az adatbázis tartalmát, akárcsak bármelyik REST kliens használata esetén).

Ehhez látogassunk el a <https://editor.swagger.io/> oldalra, majd a felső sávban, a File menün belül válasszuk az Import file lehetőséget. Válasszuk ki a \cardservice\ mappában található cardservice-swagger-doc.yaml fájlt.

Ha mindent jól csináltunk, a jobb oldali ablakban láthatjuk a dokumentációt, mely leírja a program működését.



Ha saját, alapértelmezett beállításokkal működő WebLogic szervertünkön futtatjuk az alkalmazást, a Servers alatti listában válasszuk ki a <http://localhost:7001/cardservice> lehetőséget. Ha egyedi címet szeretnénk használni, módosítsuk a szerver címét (az alsó képen pirossal bekarikázott rész) a bal oldali ablakban található kódban.

