Feladat leírása:

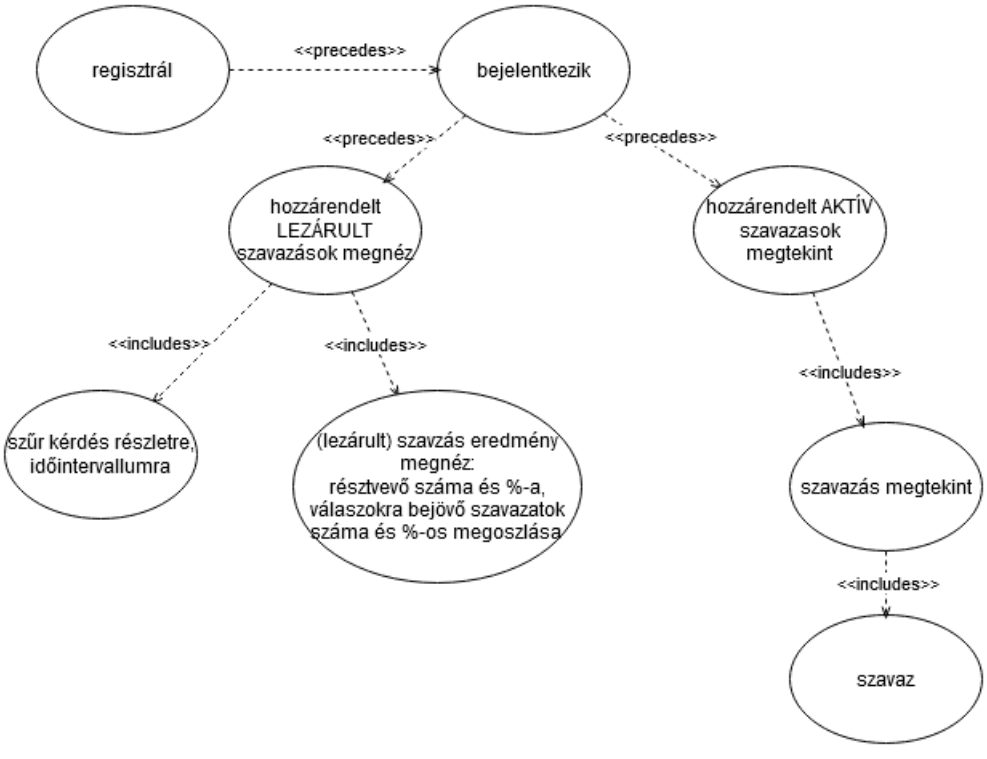
Készítsünk egy online anonim szavazó rendszert.
A kliens-szerver webalkalmazáson keresztül felhasználók meghatározott körének kérhetjük ki a véleményéta kérdés és a válasz opciók megadásával.
A szavazás titkos, amelyet a programnak garantálnia kell (külön tárolandó a szavazatukat gyakorló felhasználók és a leadott szavazat).

A webes felületen az aktív kérdésekben lehet szavazni, valamint megtekinthetőek a korábbi szavazások eredményei.

- A felhasználók email cím és jelszó megadásával regisztrálhatnak, valamint jelentkezhetnek be. A portál további funkciói csak bejelentkezést követően érhetőek el.
- Bejelentkezést követően a főoldalon megjelenik az aktív szavazások listája amelyek az adott felhasználóhoz hozzá vannak rendelve. Aktív az a szavazás, amely már elkezdődött, de még nem fejeződött be és a felhasználó még nem szavazott. A szavazásokat a befejező dátumuk szerint növekvő sorrendben kell listázni a kérdés szövegének valamint a kezdő és befejező időpontnak a feltüntetésével.
- Egy aktív szavazás kiválasztásával a weboldal jelenítse meg a kérdést és a válasz opciókat. Utóbbiak közül pontosan egyet kiválasztva lehet a szavazatot érvényesen leadni.
- A bejelentkezett felhasználók egy másik oldalon kilistázhatják a már lezárult, hozzájuk rendelt szavazásokat. Lezárultnak tekintendő az a szavazás, amelynek befejező időpontja elmúlt, vagy ha az összes hozzá rendelt felhasználó szavazott már. A szavazások listáját lehessen szűrni a kérdés szövegének részlete vagy időintervallum alapján.
- Egy lezárult szavazás kiválasztásával a weboldal jelenítse meg annak eredményét:
  - szavazó résztvevők számaés százalékos értéke;
  - válasz opciónként a szavazatok száma és százalékos értéke

Feladat elemzése:

Use Case Diagram:



Nézetek:

A _Layout elrendezőnézet megjeleníti a bejelentkezett felhasználó e-mail címét(ha be van jelentkezve), amúgy a ‘regisztráció’ és ‘bejelentkezés’ gombokat tartalmazza
Az ‘Index’ főoldalon listázódnak a bejelentkezett felhasználó aktív szavazásai. Egy szavazásra kattintva megjelennek a hozzá tartozó válaszlehetőségek. Eggy válaszlehetőségre kattintva végrehajtható a szavazás.
A ‘ClosedVotings’ oldalon a felhasználóhoz rendelt lezárt szavazások láthatók. Ezeket cím-részletre, időintervalumra lehet szűrni.

Egy lezárt szavazáshoz tartozó linkre kattintva megjelennek a hozzá tartozó válaszok és statisztikák(hány ember szavazott, ez hány százaléka az összes szavazáshoz rendelt felhasználónak, illetve a válaszokra hányan, mekkora százalékos eloszlásban szavaztak).
A ‘Login’ oldalon azonosításra szolgáló e-mail cím, és jelszó megadásával lehet bejelentkezni.
A ‘Register’ oldalon e-mail címet, jelszót és megerősítő jelszót kell megadni a regisztrációhoz.
Komponens diagram:

Kontrollerek:

‘**HomeController**’:

- [httpGet]**Index**(int? VotingId) akciója lekéri a voterService-től a bejelentkezett felhasználóhoz tartozó folyamatban lévő szavazásokat(List<Voting> típusú modellen keresztül), illetve amennyiben votingId-t kitölti a kliens, akkor a votingId-nak megfelelő szavazás részleteit ViewBag-ben átküldi.Ha nincs felhasználó bejelentkezve, akkor csak egy üres főoldalat küld vissza.
- [httpGet]**Vote**(Int32? votingId, Int32? answerId){…} átadja a paraméterben kapott votingId-t, answerId-t és a bejelentkezett felhasználó azonosítóját a ‘voterService.Vote(..)’ metódusnak, az ellenőrzést, és rögzíti a szavazást. Ha a szavazás sikeres, akkor a ‘Vote’ akció egy ‘Sikeres szavazás’ feliratú weboldalt küld vissza, amúgy egy ’Sikertelen szavazás’ feliratút. Ezen eredményt jelző weboldalakról link segítségével vissza lehet jutni a főoldalra.
- [HttpGet]**ClosedVotings**(int? VotingId) ellenőrzi, hogy be van-e jelentkezve a felhasználó. Ezután amennyiben a votingId nem null értékű, akkor viewBag-ben visszaküldi a votingId-nak megfelelő szavazás részleteit, statisztikáit egy VotingStat típusú objektumban, amit a voterService getVotingStats metódusával hoz létre. Ezen kívül a lezárult szavazások listáját elküldi ViewBag-ben. A ClosedVotings weboldalt adja vissza.
- [HttpPost]**ClosedVotings**(VotingFilter filter) a modellben kapott szűrőfeltételt ellenőrzi, és ha helyes, akkor viewBagben már csak a szűrőfeltételeknek megfelelő szavazásokat küldi vissza.

‘**AccountController**’:

- [httpPost]**Login**(LoginViewModel model) akciója ellenőrzi a ‘model’ és az accountService login() metódusával bejelentkezteti, azaz a “user” session kulcshoz hozzárendeli a felhasználó e-mailcímét. Amennyiben az accountService Login metódusa hamissal tért vissza, visszaküldjük a Login weboldalt az eddig (hibásan) kitötött ‘model’ modellel.
- **Register**(RegistrationViewModel model) az accountService Register metódusának átadja ‘model’-t és az regisztrálja a megadott felhasználót, vagy hamis értékkel tér vissza. Utóbbi esetben a Kontroller Akciója visszaküldi a ‘Register’ weboldalt ‘model’ modellel.
- **Logout**() kijelentkezteti a bejelentkezett felhasználót: a session-ből eltávolítja a felhasználó e-mailcímét.

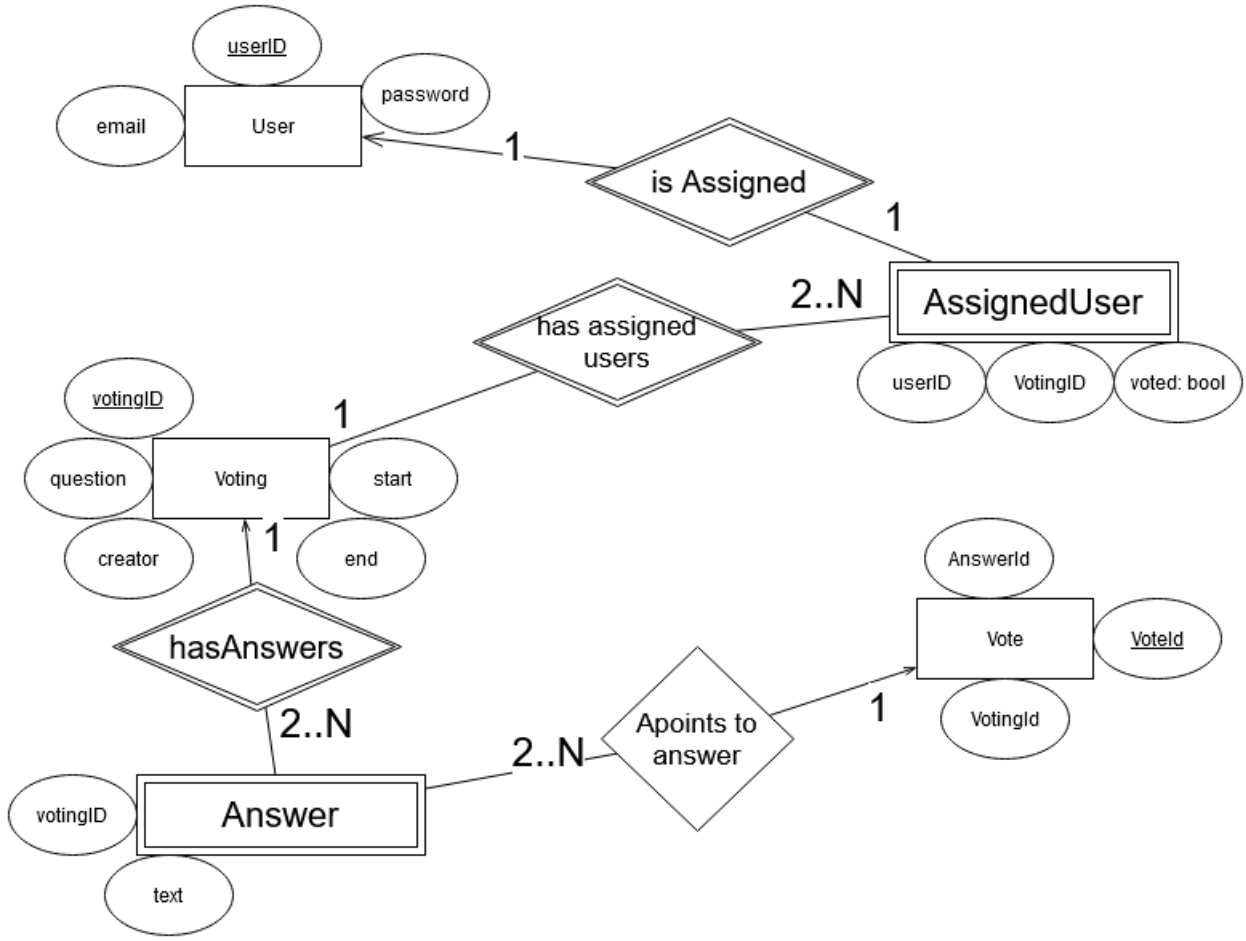
Service-ek:

- **AccountService**
  - **Login**(LoginViewModel user) sikeres validáció esetén kikeresi az adatbázis kontextusból ‘user’ emailcímének megfelelő sort, és annak jelszó oszlopát

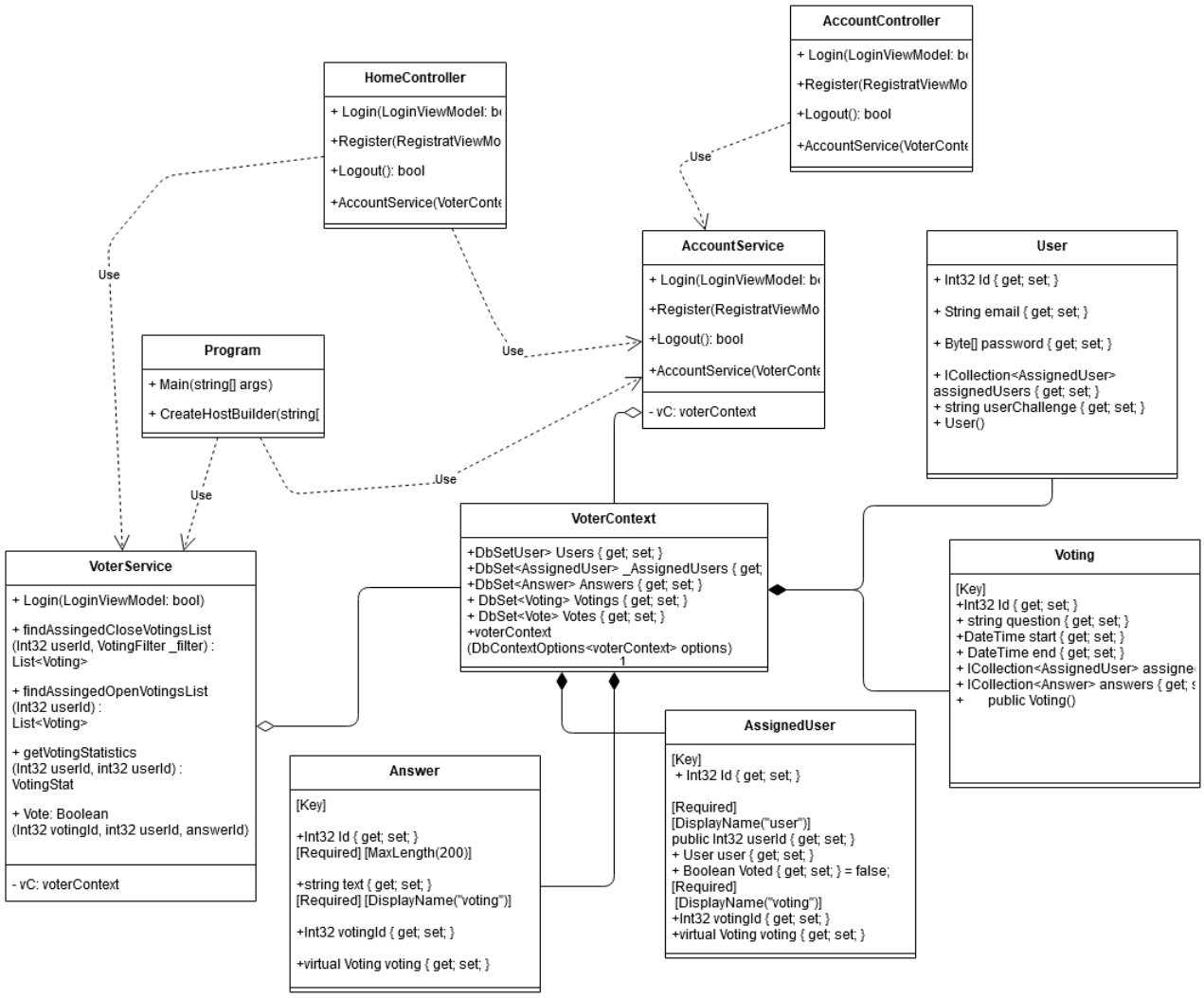
összeveti ‘user’ jelszavának a hash-elt változatával. Ha egyeznek, akkor felveszi a felhasználót sessionbe az emailcíme alapján
  - **Register(**RegisterViewModel user**)** amennyiben ‘user’ emailcíme nem foglalt, felveszi az adatbázisba a felhasználót.
  - **Logout**() kijelentkezteti a felhasználót: törli a sessionből
- **VoterService:**
  - **findAssignedClosedVotings**(Int32 userId, VotingFilter _filter) a lezárt szavazásokat szolgáltatja a HomeController.nek
  - **findAssignedOpenVotings**(Int32 userId) a folyamatban levő szavazásokat szolgáltatja a HomeController.nek
  - **Vote**(Int32? userId, Int32? votingId, Int32? AnswerId) ellenőrzés után végrehajtja a szavazást: felveszi az adatbázisba az új szavazást

Adattárolás:

Az adatbázis EgyedKapcsolat diagramja:



A program UML osztálydiagramja:



•