# Funkční požadavky

1. Do sociální sítě mohou vstupovat pouze registrovaní uživatelé.
   * **Hotovo**
2. Sociální síť umožňuje vyhledávat a zobrazovat výsledky o hledaných uživatelích.
   * **Nemáme**
   * Pouze registrovaný
   * Hledáme uživatele
   * Stačí zobrazit jeho jméno
3. Eviduje a spravuje údaje o jednotlivých uživatelích typu student (jméno, příjmení, kontakty,rok studia, obor, apod.) a o uživatelích typu vyučující (jméno, příjmení, kontakty, vyučované předměty, apod.)
   * **Hotovo**
4. Eviduje a spravuje údaje o skupinách, skupina je přiřazena vždy jednomu nebo více předmětů.
   * **Hotovo**
5. Eviduje a spravuje zaslané zprávy od jednotlivých uživatelů, přičemž u každé zprávy je nutné zaznamenávat minimálně datum a čas odeslání, odesílatele a adresáta a samozřejmě její obsah.
   * **Hotovo**
6. Zprávy lze zasílat samostatně nebo v rámci skupiny. Samostatné zprávy jsou soukromé (neveřejné) určené jen pro jednoho nebo více adresátů. Neveřejné zprávy jsou zobrazeny pouze odesílatelům a příjemcům. Veřejné se zobrazují všem uživatelům ve skupině.
   * **Hotovo**
7. Databázová aplikace umožňuje přidávat, modifikovat a mazat záznamy ve všech tabulkách minimálně pro roli administrátor.
   * **Hotovo**
   * Bude asi třeba přidat pro nové tabulky
8. Grafické rozhraní aplikace bude funkční a bude umožňovat editovat jakýkoliv záznam, který je načtený z databáze.
   * **Hotovo**
   * Bude asi třeba přidat pro nové tabulky
9. Aplikace bude mít menu nastaveno tak, že je možné z jedné karty přepnout na všechny ostatní, tak aby byla zaručena příjemná uživatelská správa.
   * **Nemáme**
   * GUI je třeba přepracovat na taby
10. Všechny tabulky musí být naplněny řádnými daty.
    * **Hotovo**
    * Bude třeba upravit pro nové taublky
11. Aplikace bude umožňovat uživateli naplnění dat pomocí webových služeb nad IS/STAG, https://stag-ws.upce.cz/ws/web, zároveň aplikace umožňuje vkládat data samostatně.
    * **Nemáme**
    * Implementace stažení, přeparsování na csv a insert
    * Insert samostatných dat
    * Pouze administrátor
12. Aplikace se skládá z hlavního okna aplikace, kde má možnost neregistrovaný uživatel procházet povolené položky menu. Hlavní okno aplikace také umožňuje přihlásit registrovaného uživatele.
    * **Nemáme**
    * Neregistrovaný zobrazí uživatele a skupiny
13. Kromě standardních záložek pro přihlášeného i nepřihlášeného uživatele si bude moci přihlášený uživatel typu vyučující prohlížet svoje předměty a k nim vytvořené skupiny, ve kterých může spravovat komentáře.
    * **Nemáme**
    * Nová implementace -> vyučující si bude moci zobrazit svoje předměty a skupiny, ve kterých jsou dané předměty
    * Spravovat komentáře ??????
14. Administrátorovi může spravovat jakéhokoliv uživatele, komentáře a obecně jakékoliv záznamy, zároveň se může přepnout (emulovat) jakéhokoliv vyučujícího a získat tak přehled správy nad objekty konkrétního vyučujícího.
    * Napůl
    * Implementace emulování vyučující

# Funkčnost DB

1. min. 10 tabulek navrženého datového modelu i s číselníky;
   * **Hotovo**
     + **Nové**
       - **Zprávy\_backup**
       - **Soubory**
2. min. 2 číselníky, v dokumentaci bude uvedeno, proč byly tabulky vybrány jako číselníky;
   * **Hotovo**
3. Každý umělý primární klíč bude mít vytvořenou sekvenci;
   * **Hotovo**
   * **Sekvence s triggerem**
4. min. 3 pohledy – logicky využité, různého typu;
   * **Hotovo**
     + Obory
     + Uživatelé
     + Studenti
     + Učitelé
     + Skupiny
     + Hodnocení
5. min. 3 funkce různého typu s odpovídající složitostí, triviální a podobné funkce nebudou uznány, každá z funkcí musí mít odlišný výstup, tedy pracovat s různými operacemi;
   * **2/3**
   * Nové
     + Hashování hesla
     + Výpočet průměru hodnocení skupiny a vrácení id skupiny s nejvyšším průměrem hodnocení
     + IMPLEMENTOVAT VYHLEDÁVÁNÍ!
6. min. 4 uložené procedury různého typu s odpovídající složitostí, triviální a podobné procedury nebudou uznány, každá z procedur musí mít odlišný výstup, tedy pracovat s různými operacemi, procedura může data zpracovávat i dávkově;
   * **Nemáme**
   * Dodělat složitější procedury
7. min. 2 triggery různého typu opět odpovídající složitostí, triviální a podobné spouště nebudou uznány;
   * **Hotovo**
   * Máme triggery na sekvence
   * Triggery validují inputy u všech tabulek
   * U uživatele hashují heslo (To je spíš funkce, ne?)
   * Při smazání zprávy se zpráva ukládá do backup tabulky
8. Aplikace bude umožňovat uložit vybraný binární obsah do databáze a následně jej i z databáze získat (a pokud se bude jednat o obrázek, tak i v rámci aplikace zobrazit). Pro tento úkol vytvořte ve svém schématu speciální tabulku. Tabulku navrhněte tak, aby kromě samotného binární obsahu umožnila uložit doplňkové informace, jako např.: název souboru, typ souboru, přípona souboru, datum nahrání, datum modifikace, kdo provedl jakou operaci.
   * **Nemáme**
   * Implementace nové tabulky na binární soubory
9. Aplikace bude využívat minimálně 3 plnohodnotné formuláře (např. ošetření vstupních polí, apod.) pro vytváření nebo modifikaci dat v tabulkách, ostatní potřebné formuláře jsou samozřejmostí.
   * Bude nové GUI

# Dokumentace

## Uživatelská

## Programová

# Rozšíření

1. ISKAM – objednávání jídla (Done – DB struktura)
2. Notifikace při příchozí zprávě
3. Obrázky do zpráv (komentářů) (Done – DB struktura)
4. Zabanování žáka na určitou dobu