Zadání projektu – ITU 2023/2024

Představte si, že: *Jste vývojáři v malé SW firmě, která právě dokončila velkou zakázku a chystá se začít pracovat na nové. Šéf vás požádal, abyste se poohlédli po nějaké aplikaci, jejíž používání by se dalo vylepšit. Máte oslovit pravidelné uživatele libovolné aplikace (mobilní, na PC apod.) ve svém okolí a analyzovat, nejen jakým způsobem danou aplikaci používají, ale jakou úlohu vlastně touto aplikací řeší a zda-li nelze tento proces pomocí nové aplikace udělat jinak. Díky tomu můžete identifikovat části aplikace, které by se daly vylepšit za účelem usnadnění používání. Případně můžete najít nějakou „díru na trhu" – něco, co by nějaká skupina uživatelů potřebovala, a na trhu ještě není. Hlavním cílem je však najít zajímavý projekt pro svůj tým, a zajistit tak náplň další práce a přísun nového „kapitálu".*

Přehled

- Projekt je řešen **tříčlenným týmem**.
- Týmy se nebudou vytvářet ani registrovat **v žádném systému**. Týmy si sestavte sami, není třeba je nikde předem oznamovat ani registrovat. Vyberte si kapitána týmu. Složení týmu je nutné uvést až do zprávy k návrhu (viz podkapitola 1.6).
- Projekt sestává ze tří částí: Návrh, Implementace a Obhajoba. Všechny části projektu jsou povinné. Při nesplnění jakékoliv z částí projektu bude projekt hodnocen 0 body.
- Návrh aplikace (1.–7. týden) sestává z výběru vhodného tématu, průzkumu uživatelských potřeb, návrhu aplikace (architektury), sepsání zprávy o návrhu, tvorby funkční kostry aplikace a odprezentování výsledků pomocí krátké pitch prezentace (podrobnosti viz kapitola 1).
- Implementace (8.–13. týden) sestává z programového řešení navržené aplikace, jejího testování a sepsání technické zprávy spolu s krátkým demonstračním videem (podrobnosti viz kapitola 2).
- Projekt je nutné obhájit. Obhajoby se budou konat v lednu (podrobnosti viz kapitola 3).
- Konkrétní termíny a časy odevzdávání povinných výstupů jsou uvedeny v IS VUT.
- Body za jednotlivé části projektu budou zadány do IS VUT najednou, a to až po Obhájení projektu.

1 Návrh aplikace (1.–7. týden)

Následující podkapitola spolu s obrázkem 1 slouží k jasnému a ucelenému přehledu toku práce na návrhové části projektu. Podrobnosti k jednotlivým klíčovým bodům jsou uvedeny v následujících podkapitolách.
Projekt začíná návrhem, jehož cílem je nalezení vhodného tématu a jeho rozpracování. Tato část řešení probíhá mezi 1. a 7. týdnem semestru. Její klíčové body jsou shrnuté v následující podkapitole a dále rozvedené v dalších podkapitolách.

1.1 Klíčové body

Téma (1.–2. týden)

1. **Každý člen** týmu vypracuje unikátní návrh tématu projektu.
2. *Všichni členové* se shodnou na jednom tématu.

Průzkum uživatelských potřeb (2.–3. týden)

3. **Každý člen** provede analýzu uživatelských potřeb a klíčových problémů zvoleného tématu.
4. **Každý člen** vyhledá jednu existující aplikaci, která řeší zvolené téma a na základě provedené analýzy uživatelských potřeb popíše přednosti a nedostatky dané aplikace.
5. *Všichni členové* se shodnou na uživatelských potřebách a klíčových problémech.

Návrh aplikace (3.–6. týden)

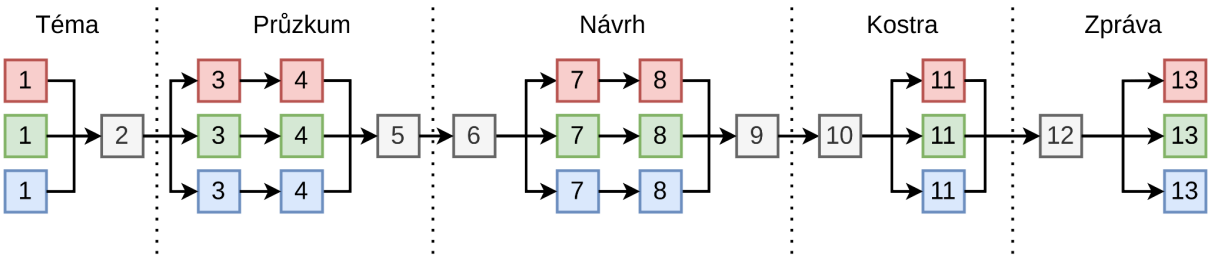
6. *Všichni členové* se shodnou na rozdělení práce mezi jednotlivé členy.
7. **Každý člen** navrhne informační strukturu a GUI své části a vytvoří maketu s ohledem na uživatelské potřeby.
8. **Každý člen** provede testování své makety.
9. *Všichni členové* se shodnou na technickém řešení a technologiích (programovací jazyk, konkrétní realizace MVC, definice API, datové struktury).

Funkční kostra aplikace (6.–7. týden)

10. *Všichni členové* společně připraví kostru aplikace.
11. **Každý člen** dopracuje svou část kostry.

Odevzdání zprávy o návrhu a pitch prezentace (7. týden)

12. *Všichni členové* připraví a odevzdají jednu identickou zprávu o návrhu, **každý člen** v ní popíše svoji práci, společná práce může být popsána *více členy*.
13. **Každý člen** vytvoří, odevzdá a odprezentuje svoji pitch prezentaci.



Obrázek 1: Diagram reprezentující tok návrhové části projektu. Paralelní barevné bloky značí **samostatnou práci jednotlivých členů týmu**, šedé bloky značí *společnou práci*, čísla odkazují jednotlivé klíčové body v podkapitole 1.1.

1.2 Téma (1.–2. týden)

ad 1) – **Každý člen** týmu vypracuje unikátní návrh tématu projektu.

- Inspirujte se již existujícími aplikacemi, svými vlastními potřebami a potřebami lidí ve vašem okolí.
- Ideální téma vychází z reálných potřeb uživatelů, kteří jsou nespokojeni s dostupnými řešeními. Tématem pak může být vylepšení nevyhovující existující aplikace nebo vytvoření aplikace úplně nové.
- Zpráva o návrhu: popište Vámi navržené téma a uveďte zdůvodnění tématu ve formě naplnění konkrétních uživatelských potřeb.

ad 2) – *Všichni členové* se shodnou na jednom tématu.

- V rámci týmu se shodněte na jednom ze tří navržených témat
- **Odteď již pracujete pouze na společně zvoleném tématu.**
- Zpráva návrhu: stačí uvést vybrané téma a zdůvodnění.

1.3 Průzkum uživatelských potřeb (2.–3. týden)

ad 3) – **Každý člen** provede analýzu uživatelských potřeb a klíčových problémů.

- Připravte si dotazník.
- Proveďte průzkum mezi potenciálními uživateli pomocí rozhovorů a Vámi navrženého dotazníku.
- Na základě průzkumu proveďte analýzu uživatelských potřeb a klíčových problémů.
- Počet dotazovaných uživatelů musí být **minimálně 2**. Uživatelé musí být relevantní, tzn. reální uživatelé dané aplikace (úředník, číšník, učitel, řidič, hudebník, kuchař apod.).
- Zpráva o návrhu: uveďte dotazník ve formě jednotlivých otázek, poznatky získané z odpovědí, analýzu těchto odpovědí s důrazem na potřeby uživatelů a klíčových problémů.

ad 4) – **Každý člen** vyhledá jednu existující aplikaci, která řeší zvolené téma a na základě provedené analýzy uživatelských potřeb popíše přednosti a nedostatky dané aplikace.

- Zjistěte přednosti a nedostatky vybrané aplikace s využitím informací z průzkumu uživatelských potřeb (bod 3).
- Na základě toho si ujasněte, jakou funkcionalitu by měla mít vaše budoucí aplikace a jaké by se naopak měla vyvarovat.
- Zpráva o návrhu: krátce popište Vámi nalezenou existující aplikaci, její přednosti a nedostatky; jak se vaše budoucí aplikace bude inspirovat přednostmi a jak bude řešit nedostatky.

Pozn. k 3,4: V rámci bodu 3 může být výhodné zjistit od uživatelů přednosti a nedostatky vybraných aplikací, tyto poznatky můžete pak přímo využít v bodu 4.

ad 5) – *Všichni členové* se shodnou na uživatelských potřebách a klíčových problémech.

- Na základě uživatelských průzkumů, analýz uživatelských potřeb, klíčových problémů a výhod/nevýhod existujících aplikací se všichni členové týmu shodnou na klíčových vlastnostech budoucí aplikace, **tyto vlastnosti musí odrážet konkrétní potřeby uživatelů, které jednotliví členové zjistili v rámci bodu 3 a 4.**
- Zpráva o návrhu: uveďte klíčové potřeby uživatelů, uživatelské procesy (jak uživatelé postupují) a klíčové vlastnosti vaší budoucí aplikace, které budou tyto potřeby řešit.

1.4 Návrh aplikace (3.–6. týden)

ad 6) – *Všichni členové* se shodnou na rozdělení práce mezi jednotlivé členy.

- Rozdělení práce lze provést dvěma způsoby:
    1. **Každý člen** pracuje na **unikátním/vlastním řešení aplikace**, výsledkem jsou 3 aplikace, jejichž vlastnosti GUI lze na závěr práce porovnat.
    2. **Každý člen** pracuje na **části aplikace**, výsledkem je jedna aplikace.

- **Každý člen** týmu musí v projektu sám autorsky realizovat/implementovat nějakou **část aplikace s GUI**. Práce v týmu slouží ke snížení režie s realizací backendu, ke vzájemné podpoře a sdílení implementačních zkušeností a především k získání zkušenosti z práce v týmu.
- Zpráva o návrhu: **jasně uveďte rozdělení práce** (kdo bude na čem pracovat) **a pro jaký způsob rozdělení práce jste se rozhodli (1 nebo 2).**

ad 7) – **Každý člen** navrhne informační strukturu a GUI své části a vytvoří maketu s ohledem na uživatelské potřeby.

- Navrhněte rozložení informace a interakce do oddílů (stránek/obrazovek/sekcí atd.) a rozložení GUI prvků v jednotlivých oddílech. Návrh informační struktury vysvětlete, tedy popište důvody, proč je GUI rozděleno právě takto, jaké jsou mezi GUI prvky logické vazby atd.
- Vytvořte maketu GUI pomocí programu Figma. Ne kompletní prototyp (ne všechny funkce 1:1 s výslednou aplikací), ale maketu (viz přednášky).
- **Návrh musí reflektovat vlastnosti aplikace, na kterých se členové shodli v rámci bodu 5.**
- Zpráva návrhu: krátce popište jak váš návrh řeší potřeby uživatelů (odkazujte se na bod 5), maketu reprezentujte pomocí obrázků, diagramů nebo snímků obrazovky a vhodného krátkého popisu.

ad 8) – **Každý člen** provede testování své makety.

- Navrhněte uživatelský experiment, který umožní ověřit, že vytvořená maketa řeší uživatelské potřeby a klíčové problémy z bodu 5.
    - Definujte metriky, které budete vyhodnocovat, a které umožní prokázat použitelnost navrženého rozhraní
    - **POZOR! Prosté tvrzení, jako „uživatelům to fungovalo“, „uživatelům se to líbilo“ nebo „uživatelé byli spokojení“, není metrika!**
    - Definujte testovací úlohy/scénáře (co budou testované subjekty při testování dělat).
    - Metrikou může být například: počet dotazů uživatele (např. jak má dále postupovat), úspěšnost dokončení scénáře/úlohy (success/error rate), počet pokusů nutných k nalezení nějaké informace (number of trials), úspěšnost nalezení předpokládané cesty při navigaci (expected path).
- S využitím navrženého experimentu proveďte testování **alespoň na 2 uživatelích**.
- Navrhněte změny, které řeší nedostatky zjištěné při testování. Tyto změny by měly být reflektovány ve výsledné implementaci (viz kapitola 2).
- Pokud testování nepřinese žádná zjištění a neodhalí nedostatky, nebylo provedeno správným způsobem, a je ho potřeba udělat jinak a znovu. Testování neslouží k potvrzení si návrhu, ale nalezení jeho nedostatků, nejasných částí apod.
- Zpráva o návrhu: popište použité metriky a proveďte jejich vyhodnocení, popište testovací úlohy/scénáře, vložte informace o testovaných subjektech, popište, jak probíhalo testování, jaké nedostatky odhalilo, a vámi navržené řešení těchto **nedostatků**.

ad 9) – *Všichni členové* se shodnou na technickém řešení a technologiích (programovací jazyk, konkrétní realizace MVC, definice API, datové struktury).

- Navrhněte technické řešení a vyberte technologie k realizaci.
- Navrhněte architekturu aplikace, použijte návrhový vzor MVC nebo obdobný (MVP, MVVM apod.).
- Specifikujte vlastnosti/chování logiky aplikace (Model), a navrhněte potřebné datové struktury, funkce a API.

- Zpráva o návrhu: popis architektury celé aplikace (části FE a BE*) s ohledem na návrhový vzor MVC (nebo obdobný), popis datového modelu (datové struktury, hlavní funkce a metody), definice API (ne všechny funkce, jen ty klíčové), seznam vybraných technologií a nástrojů (pro FE i BE) včetně stručného zdůvodnění jejich výběru.

** Vysvětlení FE a BE je v kap. 2.2.*

1.5 Funkční kostra aplikace (6.–7. týden)

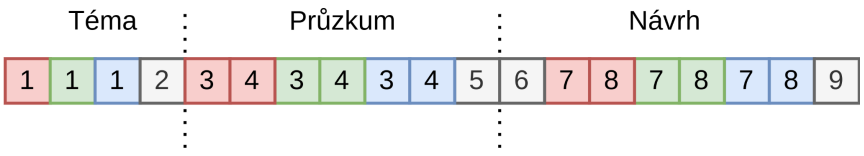
ad 10) – *Všichni členové* společně připraví kostru aplikace.

- *Všichni členové* se shodnou na způsobu spolupráce při vývoji SW aplikace (např. vytvoří repozitář na GitHubu apod.). Členové týmu mohou společně vytvořit a sdílet:
  - kostru aplikace ve vybraném vývojovém nástroji/technologii,
  - základní datový model (stačí jedna jednoduchá datová struktura),
  - minimální funkčnost BE, např. pouze operace Create a Read.

ad 11) – **Každý člen** dopracuje svou část kostry.

- **Každý člen** musí mít:
  - nainstalované vývojové prostředí a potřebné technologie k vývoji SW aplikace,
  - vlastní kopii kostry aplikace,
  - kostru doplněnou o vlastní prvek GUI a napojení na BE, např. jedno tlačítko k odeslání požadavku/zavolání funkce a textový GUI element k zobrazení výsledku.
    - V případě implementace webové aplikace je nutné využít asynchronní komunikaci.

1.6 Odevzdání zprávy o návrhu a pitch prezentace (7. týden)



Obrázek 2: Diagram reprezentující strukturu zprávy o návrhu návrhové části projektu. Barevné bloky značí samostatný text jednotlivých členů týmu, šedé bloky značí společný text více členů týmu, čísla odkazují na jednotlivé klíčové body v podkapitole 1.1.

Každý člen odevzdá archiv (xlogin00.[zip|rar|tar]) pojmenovaný svým loginem (např. xnovak01.tar). Odevzdává se do IS VUT k aktivitě „Návrh termínu“.

Obsah archivu:

- 1× zpráva o návrhu (xlogin00_zn.pdf), **stejná pro všechny členy.**
- 1× pitch prezentace (xlogin00_pitch.pdf), **různá pro každého člena.**
- 1× zdrojový kód funkční kostry aplikace, **stejný pro všechny členy** (viz kap. 1.5).

ad 12) – *Všichni členové* připraví a odevzdají jednu identickou zprávu o návrhu, **každý člen** popíše **svoji práci**, *společná práce* může být popsána *více členy*.

Rozsah a obsah zprávy o návrhu

- **Seznam všech členů týmu, označený kapitán týmu.**
- Zpráva návrhu musí sledovat strukturu na obrázku 2.
- **Každý člen** popisuje svoji práci (body 1, 3, 4, 7 a 8), společná práce (body 2, 5, 6 a 9) může být popsána *více členy.*
- Doporučený rozsah zprávy návrhu je cca 10–12 normostran, bez započtení obrázků maket. Rozsah jednotlivých bodů by měl být cca podobný (kromě bodů 2 a 6, kde stačí pouze několik vět).

ad 13) – **Každý člen** vytvoří, odevzdá a odprezentuje svoji pitch prezentaci.

Rozsah a obsah pitch prezentace

- Výsledné téma vybrané týmem (bod 2).
- Vlastní návrh na zlepšení existující konkurenční aplikace nebo novou aplikaci vycházející z průzkumu uživatele a existujících aplikací (bod 3 a 4).
- Ukázka vlastní makety (bod 7).
- Vybrané použité technologie a architektura aplikace (velmi stručně) (bod 9).
- Snímky obrazovky funkční kostry aplikace (bod 10).

Prezentace pitch prezentace proběhne v rámci cvičení 8. týden semestru. Délka prezentace je **1 minuta** + 1 minuta diskuze. Prezentace je povinná pro každého člena týmu. Příklad pitch prezentace (s rezervou 10s):

- *Dobrý den, jsem Víťa Beran. Zjistil jsem, že zvířecí útulky potřebují dobrovolníky pro venčení psů. Existující rezervační nástroje jsou nevyhovující, udělat si rezervaci dobrovolníky odradí. (14s)*
- *Dobrovolník si potřebuje na týden až měsíc označit, kdy může nebo nemůže, vybrat si druh a velikost psa a dostat informaci, kdy je potřeba jeho pomoci. Tyto termíny si chce zpracovat na mobilu, přehledně a rychle. (25s)*
- *Test na maketě navrženého GUI odhalil potřebu přidat potvrzení při změně termínu a odstranění některých informací z kalendáře, což ušetřilo místo a kalendář je přehlednější. (40s)*

- *Aplikace bude vytvořena pomocí webových technologií, konkrétně React pro FE a Django pro BE. Funkční prorotyp přidává termín a maže termín pomocí tlačítka. (50s)*
- *Děkuji.*

1.7 Hodnocení

Hodnocení proběhne cca během 9. a 10. týdne. Důležité body, které budeme kontrolovat:

Technická zpráva (až 6b)

- neobsahuje gramatické, syntaktické chyby a překlepy
- obrázky mají popisky
- informační úroveň (text není pouze „omáčka“, je stručný a k věci)
- formální kvalita (formátování textu, kvalita obrázků)
- popis rozdělení práce

Návrh uživatelského rozhraní a architektury aplikace (až 10b)

- kvalita analýzy uživatelských potřeb, popis uživatelských procesů a informační struktury
- kvalita makety ve Figmě
- kvalita analýzy existujících aplikací
- **popis klíčových částí datového modelu (datových struktur)**
- **správné použití MVC (nebo obdobného vzoru)**
- **definice API klíčových funkcí (ne všech, pouze těch klíčových)**

Testování (až 6b)

- splněn minimální počet respondentů
- řádně popsány scénáře, průběh a výsledky testování
- popis a použití metrik při testování

Tučně označené body jsou povinné a zásadní.

Odevzdání funkční kostry aplikace je podmínkou pro získání bodů za „Návrh aplikace“.

Body za jednotlivé části projektu budou zadány do IS VUT najednou, a to až po „Obhájení projektu“.

2 Implementace (8.–13. týden)

Implementace projektu může být provedena pomocí libovolné technologie na libovolném zařízení (desktop, mobilní zařízení, wearable zařízení apod.), pokud dává daná technologie a zařízení pro vybranou aplikaci smysl.

Každý člen implementuje svoji část návrhu.

2.1 Klíčové body

Všechny aplikace

1. Implementace MVC (nebo obdobného návrhového vzoru).
2. Interaktivita s uživatelem.
3. Propojení GUI komponent (změna hodnoty komponenty se automaticky projevuje v ostatních relevantních komponentách).
4. CRUD – Create, Read, Update, Delete.
5. Implementace registrace, přihlášení a správy profilu uživatele není vyžadováno a **nebude hodnoceno.**

Webové aplikace (**dodatečné** požadavky)

6. Moderní frameworky.
7. Asynchronní komunikace.

2.2 Všechny aplikace

ad 1) – Implementace MVC (nebo obdobného návrhového vzoru).

Základním technickým požadavkem na řešení je důsledné oddělení frontendu (dále jen FE, určeného ke komunikaci s uživatelem) a backendu (dále jen BE, určeného pro uchovávání dat a manipulaci s nimi) aplikace, které spolu komunikují pomocí jasně definovaného rozhranní (API). Oddělení logiky aplikace (modelu) od uživatelského rozhraní (view) je klíčové pro obhájení projektu. Inspirovat se lze například návrhovým vzorem MVC nebo obdobným (MVP, MVVC apod.).

Pozor, BE nemusí znamenat server apod. BE je principiální oddělení logiky aplikace (dat a funkcí s daty) od GUI (zobrazení a interakce). BE jsou často jen třídy (datové struktury a metody) pro správu dat a logiky aplikace, je-li potřeba, i datové úložiště. Velmi zhruba – model MVC (bude probíráno ve výuce) to právě takto specifikuje – Model (data a metody) a View (zobrazení a interakce). Controller pak mapuje akce uživatele (interakci) na konkrétní funkce Modelu a zpětně zajišťuje zobrazení výsledku do View.

Není podstatné, jakým způsobem jsou data uložena (DB, textový soubor, JSON ve zdrojovém kódu), podstatné je, aby byla uložena na BE a manipulace s nimi byla možná POUZE skrze definované API. Plně funkční DB je sice „nice-to-have“, ale v ITU se hodnotit nebude.

Data NESMÍ být uložena ani spravována v rámci FE, GUI s nimi nesmí přímo manipulovat. Není tedy možné reprezentovat stav pomocí metadat připojeným k prvkům GUI (v případě webové aplikace atributů v DOM). V případě uživatelské interakce je nutné provést zaslání zprávy BE pomocí definovaného API a následné překreslení GUI podle výsledku, který BE vrátí. Pokud BE a FE správně oddělíte, mělo by jít bez zásahu do FE změnit úložiště dat, např. z operační paměti na disk. Zároveň by mělo být možné bez zásahu do BE vytvořit zcela nové GUI uzpůsobené pro jiný typ uživatelů, např. pro nevidomé.

Příklad hra Piškvorky: stav hry není určen stavem jednotlivých komponent GUI, které reprezentují herní políčka, ale modelem umístěným na BE. V případě žádosti o změnu stavu některého políčka ze strany uživatele předá FE skrze volání API na BE požadavek, BE jej vyhodnotí (např. zkontroluje jestli nebyla vytvořena řada koleček nebo křížků vedoucí k ukončení hry, případně jestli nebylo dosaženo remízy, kdy už není možné hru vyhrát žádným hráčem atd.), BE aktualizuje data a informuje FE o výsledku. FE aktualizuje GUI podle aktuálních dat (modelu) – upraví jednotlivé komponenty.

V případech, kdy to není nutné pro smysluplný chod aplikace, není vyžadována perzistence dat mezi jednotlivým spouštěním aplikace – např. v případě tvorby hry piškvorky není třeba ukládat rozehrané ani historické hry.

ad 2) – Interaktivita s uživatelem.

Smyslem projektu v ITU je vytvořit interaktivní aplikaci, která uživateli umožňuje nějakým způsobem pracovat a interagovat s datovým modelem. Statická aplikace, která pouze zobrazuje data a žádným způsobem s nimi nemanipuluje, nebude akceptována.

POZOR! To že aplikace manipuluje s datovým modelem neznamená, že je nutné tento model někde perzistentně ukládat, viz předchozí bod.

ad 3) – Propojení GUI komponent.

Demonstrujte, že rozumíte, jakým způsobem vybraný programovací nástroj řeší propojení jednotlivých komponent GUI a jejich dynamickou aktualizaci (pokud to vybraný framework umožňuje).

Velmi zjednodušený příklad: existují dvě GUI komponenty, jedna slouží k přidání nové položky, druhá slouží pro zobrazení položek. Ve chvíli, kdy se provede akce vložení nové položky (manipulace s daty v datovém modelu), je potřeba nějakým způsobem upozornit druhou propojenou komponentu, že si má aktualizovat svůj obsah, protože proběhla manipulace s daty, které jsou pro ni relevantní. V každém moderním frameworku existují standardní metody, které toto umožňují. Ty byste měli použít a na obhajobě být připraveni vysvětlit, jakým způsobem jste je použili a jak fungují.

ad 4) – CRUD – Create, Read, Update, Delete.

V projektu je potřeba ukázat, že s daty umíte také interaktivně manipulovat a ne je pouze zobrazit. Některá z interaktivních kompomponent (nebo jejich kombinace) musí umožnit vytvořit, zobrazit a aktualizovat obsah vybrané datové struktury. Smazání (delete) je volitelné, ale vhodné (pokud v daném případě dává smysl).

POZOR! Filtrování není operací Create ani Update. Je to pouze Read s různými parametry.

ad 5) – Implementace registrace, přihlášení a správy profilu uživatele není vyžadováno a **nebude hodnoceno.**

Pokud projekt není přímo zaměřen na nějaké nové řešení pro **registraci, autentizaci či správu rolí**, **nevěnujte se** tomu, v ITU to hodnoceno nebude. Toto je vyřešená problematika a jistě to není něco, co uživatelé pro svoji činnost potřebují, a proto tomu v projektu nevěnujte pozornost. V ITU svoji energii a pozornost zaměřte na potřebnou interakci pomáhající uživateli dosáhnout svých cílů.

2.3 Webové aplikace

ad 6, 7) – Webová aplikace – využití moderních přístupů a technologií

V případě použití webových technologií je pro akceptování řešení podmínkou využití moderních frameworků (bod 6) a asynchronní komunikace mezi FE a BE, tj. asynchronní CRUD (bod 4 a 7). Není akceptovatelné odevzdat aplikaci, která bude pro úpravu jedné položky vyžadovat znovunačtení celé stránky. Taková aplikace bude hodnocena 0 body. Prostudujte si, jakým způsobem realizuje Vámi vybraný a použitý framework komunikaci s backendem. Tato znalost bude vyžadována u obhajoby projektu.

Moderní frameworky
Mezi moderní FE frameworky patří např. Vue, React apod. Lze také použít podpůrné BE frameworky/knihovny jako Django, Flask apod. Řešení postavená pouze na čistém JavaScriptu + HTML + CSS (FE) a čistém PHP (BE) nebudou hodnocena. Využití základních technologií (JavaScript + HTML + CSS, nebo WinAPI) je obsahem přednášek a cvičení, protože demonstrují, jak je GUI technicky pomocí těchto technologií realizováno. Tyto technologie jsou uvnitř (téměř) všech moderních frameworků, knihoven a nástaveb. Smyslem těchto cvičení a přednášek je Vám ukázat, co se skrývá *na dně* těchto frameworků a jak to *tam dole* funguje. Až bude za rok nový framework, abyste ho opět snadno pochopili a rychle si ho osvojili. K řešení projektu už je ale potřeba, abyste si jeden z moderních frameworků vybrali, a v rámci řešení projektu se ho naučili a vyzkoušeli ho použít.

Moderní interakce
Využijte moderních přístupů interakce. Mezi zastaralé způsoby interakce lze řadit zbytečné využívání vyskakovacích oken pro úpravu dat, využívání struktury tabulka-formulář apod. Tato zastaralá řešení většinou vychází z nevhodného postupu při návrhu GUI. Stane se tak, když začneme nejdříve definovat datové struktury (někdo to dokonce chápe hned jako tabulky v DB) a jejich vazby a k tomu automaticky přihodí CRUD. No a jak jinak má pak návrh GUI vypadat, než jako tabulka-formulář. Návrh GUI zaměřeného na uživatele by měl ale vypadat zcela jinak. Nejdříve prozkoumat a definovat, co uživatel velmi konkrétně potřebuje dělat, jaká data a operace z pohledu potřeb uživatele jsou sémanticky blízká atd. a až pak podle této analýzy a definic navrhnout GUI. Tento proces povede ke stejnému datovému modelu, ale zcela určitě k jinému návrhu GUI (viz kap. 1 a přednášky o návrhu GUI). Lze správně namítnout, že ne vždy je špatně využít třeba interakci pomocí tabulka-formulář. Tento přístup si ale budete muset náležitě obhájit (viz postup výše).

2.4 Technická zpráva a video

Každý člen vytvoří technickou zprávu o rozsahu max. 3 NS, kde jasně a přehledně shrne: téma zadání a svoji část implementace, tzn. jaké soubory jsou jeho dílem a jakou funkcionalitu v rámci aplikace zajišťují.

Každý člen vytvoří **video** o maximální délce **2 min**, které jasně a přehledně prezentuje **jeho** odvedenou práci. Video doplňte o mluvený komentář nebo titulky. Video nechť NEOBSAHUJE přihlašování do systému nebo úpravy profilu uživatele (pokud to není hlavním cílem řešeného projektu).

2.5 Odevzdání

Každý člen odevzdá archiv (xlogin00.[zip|rar|tar]) pojmenovaný svým loginem. Odevzdává ho do IS VUT k aktivitě „Realizace aplikace“.

Obsah archivu:

- 1× zpráva návrhu (xlogin00_zn.pdf, totožná s již odevzdanou zprávou v rámci návrhové části řešení projektu), **stejná pro všechny členy.**
- 1× zdrojový kód aplikace + soubor README; stejné pro všechny členy (pracujete-li na jedné aplikaci a každý tvoří nějakou její podčást), nebo různé pro všechny členy (pracujete-li každý zvlášť na 3 různých variantách aplikace).
    - Zdrojové kódy musí ve vhodné míře obsahovat komentáře a hlavičky s konkrétním autorem daného zdrojového kódu.
    - Pokud se výjimečně stane, že na jednom zdrojovém souboru pracovali dva autoři, doplňte autory do komentářů k funkcím či k částem autorských kódů.
    - Multimediální a další zdrojové soubory.
    - *readme.txt* (nebo *README.md*) s popisem instalace a odkazy na využité knihovny a jejich licence.
- 1× technická zpráva (xlogin00_tz.pdf), **různá pro každého člena.**
- 1× video (xlogin00_video.[mp4|...]), **různé pro každého člena.**

2.6 Hodnocení

Implementace (až 33b)

Při hodnocení budeme kontrolovat následující body:

- **interaktivita**
- **správné použití MVC (nebo jiný obdobný)**
- **webová aplikace používá asynchronní komunikaci**
- jasně popsáno, kdo na čem pracoval
- projekt obsahuje minimálně tyto CRUD operace **CREATE, READ, UPDATE**
- použití moderního frameworku
- k interakci s uživatelem se nepoužívají zastaralé metody
- míra rozsahu implementace (minimalistické řešení bude hodnoceno minimalisticky)
- kód je logicky členěný a je jednoduché se v něm orientovat

- kód obsahuje klíčové komentáře, autorství apod.

Tučně označené body jsou povinné a zásadní.

Body za jednotlivé části projektu budou zadány do IS VUT najednou a to až po „Obhájení projektu“.

3 Obhajoba

Každý člen týmu musí při obhajobě prokázat, že rozumí funkčnosti jeho části řešení, že ovládá klíčové aspekty zvolené technologie a že je prokazatelně využil ve své části řešení (např. v případě využití webových technologií musí každý člen ve své části použít asynchronní komunikaci).

Obhajoby projektů budou probíhat v lednovém termínu. Celková doba na jeden tým (včetně zhodnocení a vystřídání týmů) je 15 minut.

- **Obhajoba slouží k obhájení autorství řešení každého člena týmu a relevantních znalostí dané problematiky.**
- Obhajoby se účastní celý řešitelský tým.
    - Pouze osobně přítomní členové mají nárok na bodové ohodnocení.
    - V odůvodněných případech (doložených cestou studijního oddělení formou překážky ve studiu) lze dohodnout jiný termín obhajoby těm členům týmu, kteří se z vážných důvodů nemohli obhajoby zúčastnit.
- Na obhajobu se **za tým hlásí pouze kapitán** a to ve zveřejněném termínu v IS VUT v době před odevzdáním projektu.

TERMÍN <u>REGISTRACE NA OBHAJOBY</u> je v IS VUT

- Doba prezentace je stanovena na maximálně 6 minut (toto je HARD limit, po kterém bude prezentující komisí vyzván k ukončení prezentace).
- Následná diskuze obvykle trvá max. 7 minut.
- Zbylý čas je určený k přípravě dalšího týmu, vykrytí případných časových skluzů a podobně.
- Komise určená k hodnocení dané prezentace a projektu se skládá z minimálně **dvou** vyučujících.
- Hodnotí se zejména:
    - kvalita prezentace klíčových aspektů projektu,
    - schopnost odpovídat na relevantní dotazy,
    - dodržení časového limitu.