Zadání projektu – ITU 2024/2025

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“.

Rámec řešení projektu

Tato část shrnuje podstatné informace k projektu, detaily jsou uvedeny dále v dokumentu.

Řešení projektu, týmy a povinnosti každého člena týmu:

- projekt řeší 2–5členné týmy,
- členové týmu musí být přihlášeni na stejný termín cvičení,
- tým si sám vymyslí a navrhne aplikaci k řešení,
- tým, resp. členové týmu musí vypracovat všechny povinné části projektu (dle upřesnění dále),
- každý člen týmu musí vypracovat vlastní implementaci GUI ke svému úkolu,
- každý člen týmu odevzdává všechny požadované materiály (v Moodlu),
- každý člen týmu musí obhájit projekt společně s celým týmem.

Výsledné materiály k odevzdání

Obsah	Termín	Název odevzdaného souboru
Návrh	Fáze I., konec 8. týdne semestru	00_login_navrh.pdf
Zdrojové kódy	Fáze II., konec 13. týdne semestru	01_login_source.zip
Závěrečný report	Fáze II., konec 13. týdne semestru	02_login_final.pdf
Krátké video	Fáze II., konec 13. týdne semestru	03_login_video.mp4/avi

Prezentace/Obhajoba

- Kontrolní prezentace (v 9. týdnu semestru) – funkční základ aplikace
- Obhajoba (zkouškové období) – výledná implementace

Hodnocení

Průběžné hodnocení v rámci *Kontrolní prezentace*, kde bude hodnoceno:

- zadání a plán činnosti jednotlivých členů týmu,
- plnění *Správné realizace GUI*.

Výsledné hodnocení projektu

- staví na informacích o vypracování povinných částí projektu v odevzdaných materiálech a kvalitě obhajoby,
- kromě celkové práce týmu bude při hodnocení reflektována autorská činnost a znalosti každého člena týmu,
- kvalita obhajoby bude hodnocena pro každého člena týmu na základě jeho znalosti svého vlastního řešení a znalostech použitých technologií,
- výsledné hodnocení 1 projektu hodnotí minimálně 2 různí lektoři,
- výsledné bodové hodnocení bude reflektovat kvalitu vypracování všech částí projektu, od nedostatečné kvality řešení (0%), přes minimalistické řešení (50-65%), přes standardní kv. řeš. proj. (65-85%), až po excelentní kv. řeš. (85-100%),
- při nesplnění podmínek projektu (správné zapracování připomínek z *Kontrolní prezentace*, *Správná realizace GUI*, dostatečný rozsah autorské práce, prokázání znalostí při obhajobě apod.) bude autorovo vypracování projektu hodnoceno jako nedostatečné (0 bodů).

Pravidla a doporučení

Týmy

- Projekt řeší 2–5členné týmy.
    - Větší tým má výhodu jednoho BE (backendu), ale může mít problém s definicí úlohy pro každého člena. Je nutné dobře promyslet činnosti v rámci týmu tak, aby měl každý člen týmu svůj řádný a relevantní úkol (více viz Fáze I. Zadání, bod Rozdělení práce týmu na FE).
- Práce v týmu:
    - snižuje náklady na režie (tvorba BE)
    - usnadňuje první kroky při používání technologií nováčkům,
    - členové týmu si pomáhají a radí, sdílejí dobré zkušenosti,
    - přináší více pohledů na věc, více inspirace a nápadů,
    - umožňuje lepší vhled do problému, protože je k dispozici více průzkumů s uživateli.
- Práce každého autora na své vlastní části řešení je nutná k prokázání znalostí a dovedností při realizaci GUI.
- Členové týmu musí být přihlášeni na stejný termín cvičení.
    - případné výměny termínů cvičení mezi sebou si domlouvají a realizují členové týmu sami,
    - v týmu nesmí být člen z jiného termínu cvičení.
- Při řešení projektu si pohlídejte, aby byl každý člen týmu nahraditelný, vyhnete se tak situaci, kdy např. autor celé myšlenky a hlavní programátor backendu z týmu vypadne a tím je zbytek týmu paralyzován.

Autorství v týmu

- každý člen týmu musí vypracovat vlastní implementaci GUI ke svému úkolu (více viz Rozdělení práce v týmu na FE),
- pokud u některého bodu není explicitně uvedeno „*každý člen týmu*“, pak lze řešit týmově,
- pokud je explicitně uvedeno „*každý člen týmu*“, pak i v odevzdávaných materiálech a při prezentacích musí každý člen týmu jasně specifikovat své řešení/výstup,
- další detaily autorství jsou uvedeny u konkrétních podmínek odevzdávaných materiálů.

Prezentace

- mají časový limit, který je nutné dodržovat,
- předem si jasně určete, kdo bude prezentovat týmové části, ať neztrácíte čas,
- prezentaci si každý tým (jako celek) i každý člen pečlivě připravte,
- vyzkoušejte si to alespoň 2x předem, ať si dobře ujasníte a připravíte, jaké klíčové informace potřebujete sdělit a co je naopak méně podstatné.

Slovníček pojmů

- FE – frontend

    - klientaská část aplikace s GUI,
    - FE je klíčová část projektu a pouze FE část je hodnocena v projektu ITU.
- BE – backend
    - část aplikace realizující převážně datový model, výpočetní funkce apod., často server nebo knihovna funkcí,
    - tuto část lze převzít z jiného kurzu FIT, externích knihoven, služeb apod.,
    - tato BE část není hodnocena v projektu ITU.
- CRUD – Create, Read, Update, Delete
- API – Aplikační rozhraní
    - Sbírka funkcí, tříd nebo protokolů vybrané knihovny, které jsou dostupné programátorovi při tvorbě aplikace založené na vybrané knihovně.
- MVC – Model-View-Controller
    - SW architektura dělící data aplikace, uživatelské rozhraní a řídící logiku,
    - podrobněji prezentována na přednáškách.

Správná realizace GUI

Všechny aplikace

Implementace MVC nebo podobné architektury

- Základním technickým požadavkem na řešení je důsledné oddělení FE a BE 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 ekvivalentním.
- 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 zpracována v/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 aplikace 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 jiné GUI, např. GUI jiného autora reflektující jiným způsobem uživatelské požadavky nebo uzpůsobené pro jiný typ uživatelů, např. pro nevidomé.

- 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.

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.

Propojení GUI komponent

- Demonstrujte, že rozumíte, jakým způsobem vybraný programovací nástroj (technologie, framework) 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 daný framework toto řeší (funguje).

Interaktivní manipulace s daty

- V projektu je potřeba ukázat, že **s daty umíte také interaktivně manipulovat a ne je pouze zobrazit**. Některá z interaktivních komponent (nebo jejich kombinace) musí realizovat manipulaci s daty, tedy aktualizovat obsah vybrané datové struktury, popř. vytvořit nový záznam.
- POZOR! Filtrování nevytváří ani nemanipuluje s daty, ale pouze data čte s různými parametry filtrace.

Registrace, přihlášení, správa profilu, tmavý režim

- Implementace registrace, přihlášení, správy profilu a tmavého režimu uživatele není v principu potřeba pro interaktivní GUI a v projektu ITU **nebude hodnocena**.
- 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 svou činnost potřebují, a proto tomu v projektu nevěnujte pozornost. V ITU svoji energii a pozornost zaměřte na potřebnou interakci pomáhající uživateli dosáhnout svých cílů (uživatelských potřeb).
- Smyslem projektu do ITU není vytvoření nějaké aplikace jako kompletního celku (i kdyby uživatel explicitně autentizaci vyžadoval), ale relevantní návrh a vytvoření GUI částí aplikace s ohledem na kvalitní zpracování GUI.

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 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.
- Lze správně namítnout, že ne vždy je špatně využít třeba interakci pomocí tabulka-formulář. Tento přístup si ale budete muset náležitě obhájit.

Webové aplikace

Moderní frameworky a asynchronní komunikace

- V případě použití webových technologií je pro akceptování řešení podmínkou **využití moderních frameworků a asynchronní komunikace mezi FE a BE**, tj. asynchronní manipulace s daty. Není akceptovatelné odevzdat aplikaci, která bude pro úpravu jedné položky vyžadovat znovunačtení celé stránky. Taková aplikace bude hodnocena 0 body. Moderní frameworky ve většině případů řeší asynchronní komunikaci za Vás (např. React). Na obhajobě tedy bude po autorech vyžadována znalost, jakým způsobem daný framework asynchronní komunikaci provádí.
- Mezi moderní frameworky patří např. Vue, React apod. Lze také použít podpůrné frameworky/knihovny jako Django apod. Řešení postavená pouze na vlastním JavaScriptu + HTML + CSS budou hodnocena 0 body.
- 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.

Fáze I: Zadání, návrh a základ aplikace

Povinné části projektu

- Zadání + Návrh + Funkční základ aplikace

Termín

- Fáze I., odevzdání (konec 8. týdne semestru)
- Kontrolní prezentace (v 9. týdnu semestru)

Odevzdání

- Návrh v pdf (00_login_navrh.pdf)
  - společný dokument pro celý tým popisující vypracování všech povinných částí projektu v této fázi, zejména Zadání a Návrhu,
  - struktura dokumentu nechť obsahuje všechny body ze Zadání a Návrhu,
  - obsahuje vypracování úkolů jak každého člena týmu, tak týmové úkoly.

Kontrolní prezentace

- krátká prezentace v době cvičení,
- zásadní je stručně představit následující:
  - cíl/zadání projektu (slajd),
  - použité technologie (slajd),
  - jasné rozdělení práce v týmu na FE (slajd),
  - demonstrovat na živé ukázce funkčnost FE,
  - ukázat klíčové části kódu (dle požadavků lektora).
- podklady k prezentaci jsou na uvážení týmu, je doporučeno mít připraveny celkem 3 slajdy (viz body výše),
- délka prezentace je 3 min., následovat bude diskuze (viz Hodnocení níže).

Hodnocení

- během *Kontrolní prezentace* budou kontrolovány klíčové aspekty projektu, jejichž nedodržení by vedlo k nedostatečnému hodnocení Výsledné aplikace, jedná se zejména o:
  - *Správná realizace GUI*,
  - aktuálnost vybraných technologií,
  - rozsah a rozdělení práce.
- výsledkem bude slovní (a písemné) hodnocení,
- body za návrh budou přiděleny v rámci celkového hodnocení Výsledného projektu.

Zadání

Odevzdávaný report (Návrh pdf) musí obsahovat popis řešení všech bodů níže.

*Týmy si vymyslí **svoji vlastní aplikaci**, provedou průzkum konkurence a uživatelských potřeb, specifikují konkrétní požadavky na danou aplikaci a její části.*

Název a téma

- název aplikace
- členové týmu, včetně označení kapitána týmu

Uživatelský průzkum a specifikace

- kdo je konkrétní uživatel
  - *uchazeč na SŠ*
- co přesně od aplikace potřebuje
  - *naučit se nějaké téma, procvičovat si otázky*
- **požadavky uživatele** (detailní průzkum, zcela zásadní pro projekt)
  - rozhovor/pozorování/dotazování s uživatelem,
  - **každý člen týmu** toto musí provést alespoň s 1 relevantním uživatelem a shrnout svůj průzkum v Návrhu,
  - za celý tým proveďte shrnutí od všech členů týmu do jasné konkrétní detailní specifikace požadavků uživatele
    - *stačí kvízové otázky, nestačí správně/špatně, potřeba zobrazení správné odpovědi, potřeba kromě textu zobrazovat i obrázky, vidět svůj progres, vybírat si mezi různými podtématy*

Průzkum existujících řešení

- seznam existujících aplikací řešící požadovanou funkci,
- každý člen týmu zpracuje průzkum pro 2 podobné existující aplikace (různé aplikace pro tým):
  - specifikuje alespoň 3 pro/výhody/inspirace a alespoň 3 proti/nevýhody/absence každé této aplikace,
  - pro a proti nesmí být obecné (*aplikace má pěkný design, používají to všichni*, apod.), ale zcela konkrétní pro klíčové požadavky vycházející z *průzkumu požadavků uživatele*,
  - v případě velmi specifického tématu, kdy není možné najít tolik podobných aplikací, prozkoumejte aplikace, které v nějaké své části dělají něco podobného k Vašemu tématu.
- za celý tým vyvoďte z průzkumu každého člena závěry a specifikujte 3 inspirace/ponaučení pro vlastní návrh.

Zadání

- popis požadované cílové aplikace,
- včetně klíčových částí FE

Rozsah výsledné implementace FE by měl být rozsahem pro jednoho člena týmu podobný následujícím příkladům:

- správa M–N vazeb (správa skupin uživatelů, správa soukromé knihovny, nákupní košík, vybavení v místnostech, obsazení hotelových pokojů, plánování událostí v kalendáři, plánování různých variant rozvrhů apod.)
    - datový model: 2–4 tabulky, který může být stejný pro všechny členy týmu,
    - funkce: zobrazení položek, vyhledávání a výběr (filtrace) položek, přidávání položek z výběru do seznamů (skupiny uživateků, položky v seznamech, předměty v rozvrzích, položky v košíku), správa položek v seznamu/seznamech, statistické informace o položkách v seznamech (počet, suma, průměr apod.)
    - GUI: obrazovka s GUI elementy realizující přiřazování položek do seznamu (seznamů), včetně filtrování položek, odstraňování položek ze seznamu, realizace hromadných operací (více položek do seznamu najednou, jedna položka do více seznamů), automatická aktualizace a efektivní zobrazení statistik apod.
- kalkulačka, různé 2D hry (piškvorky, sudoku, dostihy a sázky apod.)
    - jedna herní pocha s interakcí
    - např. u her typu sudoku je velký prostor pro experimenty s různým způsobem zadávání hodnot do polí, pak může každý člen zkusit jiný přístup (drag&drop, výběr čísel v okolí pole pomocí gesta, výběr čísla a výběr pole atd.)
    - u složtějších her (Monopoly, Dostihy a sázky apod.) je pak prostor pro rozdělení si úkolů podle různých interaktivních částí herní desky
    - vždy je potřeba dodržet pravidlo, že každý autor musí realizovat takovou část GUI, kdy se interakcí manipuluje s datovým modelem a aktualizují se různé prvky GUI.

Rozdělení práce týmu na FE

Možnosti rozdělení práce členů v týmu na FE (implementační práce BE se v projektu do ITU nehodnotí):

- je-li aplikace rozsáhlejší, např. má více částí vyžadující M–N vazby (nebo jiné potřeby), je ideální, když každý autor realizuje jednu část,
- má-li aplikace pouze jednu funkčnost a nelze dobře vymyslet další části (víz příklady výše),
    - může každý člen týmu navrhnout a realizovat stejnou funkční část aplikace po svém, lze vymyslet více variant interakce, tyto různé varianty zrealizovat a otestovat a pak vyhodnotit, která má kde své výhody a nevýhody
    - může každý člen týmu zrealizovat navržené GUI v jiné technologii
    - v nejhorším případě pak mohou dělat všichni totéž GUI ve stejné technologii, ale zde je zcela klíčové pak prokázat autorství káždé realizace, v případě podobného řešení budou autoři při obhajobě detailně vyzkoušeni ze znalosti svého kódu a použité technologie, v případě nedostatečných znalostí a pochopení bude řešení vnímáno jako plagiátorství,
- má-li aplikace více typů uživatelů (skladník – obchodník, šéf směny – pracovník, plánovač logistiky – řidič), pak je ideální, když každý autor realizuje vlastní část pro svůj typ uživatele, větší tým pak může použít dělení viz výše.

Návrh

Odevzdávaný report (Návrh pdf) musí obsahovat popis řešení všech bodů níže.

Návrh GUI

- každý člen týmu zpracuje návrh své části GUI např. ve Figmě,
- včetně popisu **logické implikace navrženého GUI** – jak navržené GUI/jeho prvky/informační struktura atd., řeší/odráží *požadavky uživatele*,
- do reportu vložte screenshot Vaší klíčové části FE.

Výběr technologií

- jak pro tvorbu FE, tak případně BE nebo použité existující knihovny/služby,
- uveďte zdůvodnění výběru.

Návrh API k BE (v rámci týmu)

- při návrhu architektury **dbejte na oddělení GUI (FE) a logiky** (BE, aplikační logiky, modelu), vzor MVC (nebo ekvivalentní),
- navrhněte a popište API (klíčových funkcí) vlastního BE,
    - případně popis API použité existující knihovny/služby
- popište klíčové datové struktury modelu,
- popište napojení FE na BE (GUI na API),
    - např. která funkce API dodá prvku GUI data k zobrazení, nebo zpracuje akci (klik) uživatele, nebo zpracuje zadaná vstupní data a jaký dodá výstup atd.

Funkční základ aplikace

Funkčním základem aplikace se myslí fungující aplikace obsahující zatím základní části GUI realizující podstatné/klíčové funkce aplikace. Každý autor musí mít funkční základ svého řešení, své části aplikace.

Implementace BE

- instalace technologií pro BE,
- příprava dat, vytvoření dat. modelu a naplnění daty,
- implementace API.

Implementace funkčního základu FE (klienta s GUI)

- instalace technologií pro FE,
- vytvoření kostry aplikace,
- napojení na API,
- každý člen týmu vytvoří podstatné části/elementy GUI své části aplikace demonstrující funkčnost API a prokázání **dodržení *Správné realizace GUI***.

Fáze II. Výsledná aplikace (klient s GUI)

Povinné části projektu

- Výsledná implementace + Testování

Termíny

- Fáze II. (konec 13. týdne semestru)
- Obhajoba (zkouškové období)

Odevzdání

Zdrojové kódy (01_login_source.zip) výsledné implementace v archivu zip (společné pro celý tým):

- archiv musí obsahovat *readme.txt*, kde je popsána adresářová struktura s důrazem na informaci umístění klíčových částí FE a autorství konkrétních částí,
- zdrojové kódy musí být pečlivě komentovány (včetně záhlaví) s důrazem na autorství jednotlivých zdrojových souborů, resp. částí zdrojových kódů,
- archiv musí obsahovat pouze vlastní jádro řešení, pouze zdrojové soubory klíčové pro jádro aplikace, nesmí obsahovat žádné binárky třetích stran, soubory vzniklé při překladu projektu apod.

Závěrečný report v pdf (02_login_final.pdf), společný pro celý tým, obsahující:

- cíl/zadání projektu, případně zdůraznění odchylek od původního zadání,
- rozdělení práce členů týmu na Výsledné implementaci (pouze FE),
- informace o zapracování případných připomínek z Kontrolní prezentace,
- informace o provedení a výsledcích testování (viz Povinné části níže).

Krátké video (03_login_video.mp4/avi), vytvoří a odevzdá každý člen týmu sám:

- ukázka chování GUI vlastní konkrétní části aplikace, včetně audio-komentáře,
- délka videa max. 90 vteřin.

Obhajoba

- je povinnou součástí realizace projektu,
- proběhne ve zkouškovém období v termínu cvičení,
- členové týmu musí představit následující:
    - zadání, popř. odchylky od původního zadání,
    - ukázka funkčnosti jednotlivých autorských částí/řešení aplikace včetně ukázek klíčových částí zdrojových kódů,
    - zdůraznění zapracování případných připomínek z *Kontrolní prezentace*,
    - shrnutí testování a ponaučení z testování.
- podklady k prezentaci jsou na uvážení týmu, kromě povinné funkční ukázky aplikace je doporučeno mít připraveny slajdy,
- délka – maximálně 3 min. na tým + 2 min. na člena (tj. např. 9 min. pro 3 členný tým),
- následovat bude diskuze a dotazy lektorů,
- výsledkem je slovní i bodové hodnocení lektorů, i s ohledem na zapracování doporučení z hodnocení fáze I. a především na znalosti vlastního řešení a vybrané technologie.

Hodnocení (viz hodnocení Projektu)

Výsledná implementace

- finální implementace projektu a všech autorských částí
    - povinné je zapracování připomínek z Kontrolní prezentace
- v reportu musí každý autor stručně popsat:
    - jaké části FE sám implementoval,
    - odkazy do adr. struktury na vlastní zdrojové soubory, popř. jejich části.

Testování

- každý člen týmu provede testování výsledné aplikace s uživatelem, se kterým na začátku projektu dělal *průzkum požadavků uživatele*,
- je nutné nechat uživatele s aplikací pracovat a podpořit ho, aby nahlas komentoval:
    - co mu jde hlavou, chce udělat, čím si není jistý,
    - jak co v GUI chápe/nechápe
- při testování neradit a především sledovat:
    - kdy neví/není si jistý, co má udělat,
    - kdy neví/není si jistý, co se stane, když něco udělá/klikne/přejde na jinou obrazovku …
    - co se mu s aplikací dělá dobře a co ne,
    - co v aplikaci nepoužil, nepochopil …
- není třeba zjišťovat:
    - jak se mu líbí barvy nebo grafický design
- v reportu každý autor musí popsat
    - kdo byl jeho uživatel (věková skupina, pohlaví, technická zdatnost, pracovní zaměření apod.),
    - jak test probíhal,
    - 2 vybrané klíčové výsledky z pozorování (Vaše ponaučení).

FAQs

Mohu použít backend/projekt z IIS?

Ano, ale dejte pozor na to, že GUI ze zadání v IIS většinou cílí spíše na obsluhu BE, než na efektivní používání GUI uživatelem. Kvůli tomu pak většina GUI reflektující přesné znění zadání IIS projektu často sklouzne k využívání zastaralých způsobů interakce (struktura tabulka-formulář pro úpravu dat, zbytečné využívání vyskakovacích oken pro úpravu dat apod.), což je pak v ITU těžko obhajitelné. Oproti IIS je v ITU potřeba zaměřit proces návrhu GUI *na uživatele* (analyzovat uživatelské potřeby, na jejich základě navrhnout a vytvořit GUI, které bude interaktivním způsobem pomáhat uživateli dosáhnout svých cílů), nikoliv na *informační systém*.

Mohu vytvořit aplikaci pro Android/iOS?

Ano, můžete vytvořit cokoliv pro jakoukoliv platformu (mobilní / webové / desktop aplikace), bude-li to náležitě splňovat body zadání.

Lze psát odevzdávané reporty v angličtině / slovenštině?

Ano.

Je možné projekt před obhajobou dopracovat?

Ne. Projekt před obhajobou nesmí být dopracováván a musí být prezentováno odevzdané řešení.

Kolik tabulek musí mít datový model? (nové)

V příkladu u Zadání je uvedeno “správa M–N vazeb, … datový model: 2–4 tabulky”. Každý člen tedy musí mít datový model s 2-4 tabulkami.
Příklad ukazuje, jakého rozsahu by měl být nějaký modul s GUI pro interaktivní manipulaci s daty. Backend může být (a měl by být) sdílen všemi členy týmu. Počet tabulek v BE nemusí odpovídat rovnici: počet_tabulek = počet_členů_týmu * 4. Rozsah práce každého člena týmu by měl být podobný, jako v příkladu interaktivní manipulace s daty s M-N vazbami.