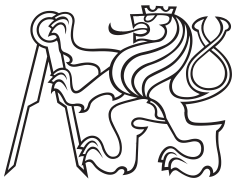
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Centrum znalostního managementu

Aplikace pro podporu metody Pomodoro

Adam Lipowski

Vedoucí: Ing. Pavel Náplava, Ph.D.
Obor: Softwarové inženýrství a technologie
Květen 2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Lipowski** Jméno: **Adam** Osobní číslo: **474425**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Softwarové inženýrství a technologie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Aplikace pro podporu metody Pomodoro

Název bakalářské práce anglicky:

Pomodoro Method Application

Pokyny pro vypracování:

Analyzujte pojmy a metody, spojené s řízením osobní produktivity a časového plánování. Zaměřte se především na metodu Pomodoro. Proveďte rešerši existujících řešení (nástrojů) a porovnejte je. Na základě výstupů provedené rešerše a analýzy metod navrhněte a vytvořte novou aplikaci, která umožní jednoduše a multiplatformově metodu Pomodoro využívat. V aplikaci reflektujte využití v akademickém prostředí, tj. využití studenty, případně vyučujícími. Funkčnost aplikace ověřte formou uživatelského testování na vybrané skupině studentů, případně vyučujících.

Seznam doporučené literatury:

[1] Francesco Cirillo, Technika Pomodoro, Jan Melvil publishing, 2019, ISBN: 978-80-7555-069-9
[2] Cal Newport, Hluboká práce: Pravidla pro soustředěný úspěch v roztěkaném světě, Jan Melvil publishing, 2016, ISBN: 978-80-7555-008-8
[3] David Allen, Mít vše hotovo (Umění produktivity bez stresu), 2016, Jan Melvil publishing, ISBN: 978-80-7555-000-2

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Pavel Náplava, Ph.D., katedra ekonomiky, manažerství a humanitních věd FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **13.02.2020** Termín odevzdání bakalářské práce: **22.05.2020**

Platnost zadání bakalářské práce: **30.09.2021**

Ing. Pavel Náplava, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. | Datum převzetí zadání

Podpis studenta

Poděkování

Hlavní díky patří vedoucímu mé práce Ing. Pavlu Náplavovi, Ph.D., za jeho zkušené vedení, vřelý přístup, ochotu a dobré nápady. Také děkuji všem uživatelům za čas, který věnovali testování aplikace a za jejich zpětnou vazbu.

Prohlášení

Prohlašuji, že jsem tuto práci vypracoval samostatně a že jsem uvedl všechny použité zdroje a literaturu.

V Praze, 19. května 2020

Adam Lipowski

Abstrakt

Cílem této práce bylo vypracovat analýzu pro návrh multiplatformní aplikace na bázi techniky Pomodoro, aplikaci následně implementovat a otestovat. Aplikace s názvem Felodoro se od všech existujících aplikací odlišuje její schopností fungovat na více zařízeních synchronizovaně. Vyjma informací týkajících se přímo aplikace Felodoro práce také obsahuje analýzu principů techniky Pomodoro a její srovnání s ostatními time managementovými technikami.

Klíčová slova: Časový management, prokrastinace, omezení vyrušování, soustředění, odhad náročnosti práce, technika Pomodoro, Node.js, MongoDB, Mongoose, GraphQL, Vue.js, Nuxt.js, PWA

Vedoucí: Ing. Pavel Náplava, Ph.D.

Abstract

The aim of this bachelor's thesis was to analyze, design and develop cross-platform application based on the Pomodoro technique and conduct user testing. Application named Felodoro is unique thanks to its cross-platform capability. This thesis also includes information about the Pomodoro technique itself. How it works, why it works and how well does it stand against other time management techniques.

Keywords: Time management, procrastination, minimizing distractions, concentration, work duration estimation, Pomodoro technique, Node.js, MongoDB, Mongoose, GraphQL, Vue.js, Nuxt.js, PWA

Title translation: Pomodoro Method Application

Obsah

Obrázky

Tabulky

Kapitola 1
Úvod

Cílem práce je analyzovat metody určené k zlepšení osobní produktivity a časového plánování a na základě analýzy navrhnout, implementovat a otestovat aplikaci pro podporu osobní produktivity. Práce pojednává především o time managementové technice zvané Pomodoro.

K vytvoření této práce mě přivedl fakt, že jsem bývalým uživatelem jedné z aplikací pro řízení osobní produktivity, a to aplikace Forest – Stay Focused. Líbila se mi představa vytvoření multiplatformního systému implementujícího techniku Pomodoro, neboť takový software zatím neexistuje.

Nejprve se práce zabývá představením time managemenetu a techniky Pomodoro, spolu s pojmy, které se v této oblasti používají. Největší část práce se zaměřuje na techniku Pomodoro, kde se používá a jaké jsou její principy. Dále práce srovnává time managementové metody a existující řešení na bázi techniky Pomodoro.

Praktická část práce obsahuje analýzu pokrývající požadavky pro multiplatformní řešení, návrh aplikace vhodné pro akademické prostředí a řešerši vhodných technologií pro její implementaci a nasazení.

Poslední kapitoly popisují testování, vyhodnocení vzniklé aplikace a směr, kterým se bude dále ubírat.

Část I

Teoretická část

Kapitola 2
Úvod do techniky Pomodoro

2.1 Seznámení s technikou Pomodoro

„Pamatuj, čas je hráč, jenž nepodvádí sice, a přece vyhrává, jak psáno, stůj co stůj," uvedl ve své sbírce Květy zla francouzský básník Charles Pierre Baudelaire. Dle mého názoru není jediný, jehož představa o plynutí času se podobá zápasu odsouzenému k porážce. Dimenzionální představa času jako měřitelné veličiny na ose mezi dvěma body (začátkem a koncem) bývá spojována s pocity úzkosti. Jeho odvíjení je prchavé a nekonečné. Francesco Cirillo ve své knize Technika Pomodoro napsal: „Snažíme-li se s během času měřit, připadáme si nedostateční, utiskovaní, zotročení, s každou další vteřinou víc a víc."

Cirillo se zaobíral negativním vlivem myšlenek typu „Uběhly už 3 hodiny, ale pořád jsem nic nestihl." a řešením je dle něj alternativní pojetí, kde čas je pouze sled událostí. Věří, že pravidelná posloupnost činností může být pro mnohé podstatou uklidňujícího rytmu. Toto smýšlení vedlo k prvnímu natáhnutí kuchyňské minutky ve tvaru rajčete, až nakonec sestavil nejpopulárnější time managementovou techniku na světě, Pomodoro[1].

2.2 Co je to Pomodoro

Pomodoro je jednoduchým nástrojem pro zlepšení produktivity (time managementová technika). Jeho cílem je minimalizování nesoustředěnosti, prokrastinace, demotivace a pocitu úzkosti spojeného s odvíjením času. Umožňuje najít udržitelné a příjemné pracovní tempo.

2.3 Princip fungování techniky Pomodoro

Fungování techniky Pomodoro je založeno na jednoduchém principu střídání časových intervalů práce a volna. Dvacet pět minut čisté práce a pět minut pauzy představují jedno pomodoro. Platí pravidlo, že pracovní interval je nedělitelný. Nelze odpracovat jen polovinu a jeho přerušení ho činí neplatným. Po práci přichází na řadu pět minut pauzy. Během ní je důležité nemyslet

na práci nebo neprovozovat jinak psychicky náročné aktivity. Umožníme tím tak naší mysli zregenerovat a získat cenný odstup a nadhled. Po čtyřech pracovních intervalech obměníme krátkou pauzu za půl hodinovou přestávku. Můžeme jí vyplnit posezením u kávy, procházkou nebo konverzací s kolegy. Opět platí, že jde o regeneraci.

Fungování Pomodora ilustruje následující obrázek:

Obrázek 2.1: Pomodoro – vizualizace techniky, zdroj: [2]

2.4 Zdůvodnění fungování techniky Pomodoro

Technika Pomodoro staví na střídání krátkých časových intervalů odpočinku a práce. Stanovení krátkého pracovního intervalu vzešlo z následujících myšlenek. Dlouhá pracovní doba působí dojmem, že je pořád na všechno spousta času a zhotovit práci můžeme kdykoliv, zatímco vytyčení pouhých 25 minut vynucuje, abychom byli produktivní okamžitě. Krátký úsek také vyvolává menší negativní emoce (nechuť, averzi) a je tak snadnější pustit se do práce.

Dalším důvodem je, že práce vyžaduje soustředění, ale udržování koncentrace stojí sílu. Sílu čerpáme z naší schopnosti ovládat se, našeho kognitivního zdroje. Pauza dlouhá 5 minut, při které nemyslíme na práci a neděláme aktivity náročné na koncentraci, poslouží k doplnění kognitivního zdroje. Ten je třeba obnovovat, neboť lidský mozek většinou udrží pozornost 20 až 30 minut, než začne myšlenkami směřovat jinam[3]. Důležitým konceptem Pomodora je také to, že regenerujeme schopnost soustředit se preventivně, ne až když je to nutné.

2.5 Závěr kapitoly

V této kapitole byly rozebírány myšlenky, které vedly ke vzniku techniky Pomodoro. Poté došlo k představení techniky samotné. Technika Pomodoro byla definována a byly zmíněny její hlavní benefity. Na konec kapitoly byl umístěn popis, jak se Pomodoro používá a proč funguje.

Cílem kapitoly bylo, aby si čtenář vytvořil obrázek o tom, co to Pomodoro vůbec je, než si přečte práci, která se touto technikou zabývá.

Kapitola 3
Seznámení s pojmy

V úvodu této práce je třeba nejprve vydefinovat pojmy, které jsou spjaté s technikou Pomodoro. Jedná se o time management, prokrastinaci, rozhodovací paralýzu, mělkou práci a hlubokou práci.

3.1 Time management

Time management je sadou postupů a praktik pro efektivní využití dostupného času. Týká se především využití pracovního času, jak jednotlivce, tak i celého projektu. Time management může být považován za podmnožinu projektového managementu a představuje plánování a sestavování harmonogramu[4].

O organizaci času mají zájem zejména lidé usilující o zvýšení osobní produktivity, ale týká se téměř všech. Stephen Covey[5] jako nejzákladnější formu time managementu považuje vytvoření upomínky na určitý čas. Jako vyšší formu vidí plánování pomocí kalendáře nebo diáře a ještě výše stojí plánování, stanovení priorit a řízení pomocí specializovaného nástroje (organizér, počítač).

3.1.1 Pilíře time managementu

Time managementové techniky obecně staví na dovednostech, které svému uživateli mají vštípit a naučit ho používat je, a jsou jimi:

- Stanovení SMART cílů.
- Překonání prokrastinace.
- Plánování (motivace).
- Zvládání stresu.
- Omezení vyrušování.

Z kapitol 4 a 6 vyplývá, že Pomodoro je rovněž postaveno na těchto pilířích.

3.2 Prokrastinace

Prokrastinace (lat. pro-crastinus = patřící zítřku) je vědomé odkládání úkolů a povinností nehledě na negativní následky[6]. Projevuje se tak, že když prokrastinujeme, nejsme schopni donutit se k vykonávání úkolů, jimž bychom se měli věnovat. Místo toho trávíme čas neproduktivními aktivitami jako například: sledováním sociálních sítí nebo seriálů, nadbytečným uklízením, hraním videoher nebo pouhým zíráním do prázdna. Jelikož ale žádná z těchto aktivit ke splnění úkolu nevede, přicházejí výčitky svědomí a frustrace vedoucí k pocitu bezmoci, který způsobí opětovné odložení povinností.

Důležité je nezaměňovat pojmy lenost a prokrastinace. Zásadně se liší tím, že líná osoba záměrně nic nedělá, dělat nechce a je takto spokojená. Naproti tomu člověk podléhající prokrastinaci by něco dělat chtěl, není však schopen se k tomu donutit[6]. Prokrastinace někdy může být také alibisticky považována za odpočinek. Ten se vyznačuje znovunabytím sil a energie, čímž nás připraví na další práci. Zmiňované výčitky a frustrace plynoucí z prokrastinace nám však z energie ubírají, a to pouze přispívá k dalšímu odkládání práce.

Prokrastinace je sice poměrně nový pojem(cca. rok 1540)[7], neznamená to však, že by lidstvo tímto neduhem trpělo až v dnešní moderní době. Dále také neplatí, že by chorobné odkládání zhoršovalo pouze náš profesní život, neboť totéž platí i pro život osobní. Lidé spíše litují věcí, které neudělali, než těch, které udělali[6]. Boj s prokrastinací sahá až k přelomu našeho letopočtu, kdy římský stoický filosof Seneca formuloval větu: „Zatímco ztrácíme svůj čas váháním a odkládáním, život utíká“.

3.3 Rozhodovací paralýza

Po vydefinováni pojmu prokrastinace můžeme postoupit k dalšímu fenoménu úzce souvisejícímu s odkládáním a tím je rozhodovací paralýza. Zmiňovaný citát filosofa Senecy ukazuje, že s prokrastinací se lidstvo potýkalo odjakživa. S rozhodovací paralýzou je to však trochu jinak. Oproti minulosti je průměrná délka živote výrazně větší, svět už se tolik nezmítá v konfliktech, hranice pro cestování téměř vymizely a v kapse díky internetu nosíme největší knihovnu na světě s výkonem větším, než měly superpočítače dvacet let zpátky. Každým dnem se zvětšuje potenciál, který svět skýtá.

Smýšlení dnešní společnosti se ubírá stejným směrem rozšiřování potenciálu. Oproti minulosti, kdy lidská uskupení žila v rigidních totalitách nebo monarchiích, je nynější ideál založen na individuální svobodě a volnosti. Panuje obecné přesvědčení, že čím více svobody lidé dostanou, tím budou šťastnější[6].

Jestli je tedy potenciál větší a větší, neměli by být lidé stále šťastnější? Intuitivně cítíme, že tomu tak není a výzkum Rotterdamské univerzity to potvrzuje[6].

Odpovědí na tento paradox je pojem rozhodovací paralýza. Jedná se o jev, kdy širší potenciál přináší více možností volby. Avšak čím více možností máme, tím je obtížnější se pro konkrétní možnost rozhodnout. Samotné vybírání té

správné možnosti vyžaduje tolik energie, že nezřídka kdy nakonec ani žádnou nevybereme. Dokonce platí, že čím je porovnávání variant složitější, tím větší je šance, že rozhodnutí pro jednu z nich bude odloženo[6].

Nejen že větší počet možností znepříjemňuje proces vybírání, má dokonce vliv také na spokojenost s vybranou variantou. Lidé si začínají představovat, jaké by to bylo, kdyby se rozhodli jinak a budou tak snáz vidět nedostatky na tom, pro co se rozhodli[6].

Zvyšující se míra rozhodovací paralýzy napomáhá růstu prokrastinace[6]. Autor knihy Konec prokrastinace píše:"Odkládání [kvůli rozhodovací paralýze] dovede snížit naši produktivitu až na zlomek reálných možností. Vědomí toho, že náš potenciál nenaplňujeme, později vede k výčitkám a furstraci."[6]

3.4 Hluboká práce

Hluboká práce je termín, který jsem převzal z knihy Hluboká práce – Pravidla pro soustředěný úspěch v roztěkaném světě. Cal Newport (autor) ji definuje jako pracovní činnost provozovanou ve stavu nerušeného soustředění, která vás nutí využívat váš kognitivní potenciál na maximum. Taková činnost vytváří nové hodnoty, zlepšuje vaše schopnosti a je obtížně replikovatelná[6].

Cal Newport v jeho knize většinou zmiňuje hlubokou práci jako dlouhý úsek věnovaný pouze práci. Pro kontext Pomodora jako hlubokou práci budu označovat soustředěné úsilí vynaložené během pracovního intervalu, napři při studiu.

Zvyk hluboce pracovat je sdílený napříč celou řadou osobností[8], které svým myslitelským úsilím ovlivnily dnešní svět. Michel de Montaigne pracoval ve vlastní soukromé knihovně, vybudované v jeho zámku. Mark Twain vytvořil Dobrodružství Toma Sawyera v chatce tak vzdálené, že jej rodina volala k obědu troubením na trubku. Carl Jung si sám postavil dům na vesnici, aby se v místnosti určené jenom pro něj mohl zaobírat svými myšlenkami a vytvářet revoluční psychologická díla. Režisér Woody Allen se stranil elektronického vyrušování a vyprodukoval 44 filmů. V odmítání počítačů s ním byl zajedno fyzik Peter Higgs, jehož izolovanost od virtuálního i okolního světa vyústila v úsměvnou situaci, kdy po oznámení udělení Nobelovy ceny jej nebyli novináři schopni ani najít.

Přínosy hluboké práce pokračují i na komerční scéně. J. K. Rowlingová při sepisování knih Harry Potter nebyla na sociálních sítích (to je na dnešní dobu poměrně významná dávka izolace a oddání se práci). Dokonce i Bill Gates praktikoval týdny přemýšlení ("think weeks"), kdy se odřízl od světa a nedělal nic, než že četl a přemýšlel.

Sklon velkých osobností hluboce pracovat vyobrazuji proto, abych ukázal jeho kontrast s pracovními návyky velké části studentů a znalostních pracovníků, které jsou od hluboké práce stále více vzdálené. Většinou mají tříštění pozornosti na svědomí síťové nástroje jako email, SMS a sociální sítě.

Jelikož se jedná o práci vytvořenou pro studenty a akademickou půdu obecně, jako hlubokou práci budu tedy označovat i studium.

3.5 Mělká práce

Mělká práce jsou kognitivně nenáročné úkony logistického charakteru, často vykonávané nesoustředěně. Tyto činnosti většinou nevytvářejí ve světě hodnoty a jsou snadno replikovatelné[6].

Společnost McKinsey na základě její studie uvádí, že běžný znalostní pracovník stráví přes 60 procent pracovního týdne elektronickou komunikací a vyhledáváním na internetu, konkrétně 30 procent času připadne na emailování[6]. V dnešní době síťových nástrojů lidé stále častěji nahrazuji hlubokou práci za mělkou[6].

Mělká práce není tak hodnotná jako ta hluboká. Neznamená to však, že je vhodné se od ní naprosto odstřihnout a považovat ji za bezcennou. Téměř každý znalostní pracovník se jí musí aspoň do určité míry věnovat. Pracovní náplň se málokdy zcela obejde bez meetingů, emailů nebo telefonátů. Jde tedy o její minimalizaci, ne úplné odstranění.

Dalším faktorem obhajujícím mělkou práci je omezenost naších kognitivních sil. Intenzivní soustředění zásobárnu těchto sil vysává. Psychologický výzkum se jal zjistit, jak dlouho tato zásoba při práci náročné na soustředění vydrží[8]. Jeho výsledky uvádějí, že u začátečníků se doba pohybuje okolo jedné hodiny za den. To není nic překvapivého. Dítě nebo člověk odvyklý na kognitivně náročnou práci začne brzy upadat na pozornosti a výstupy soustředění budou klesat. Zajímavější však je, že i osoby zvyklé na dlouhé soustředění jsou jej schopny vydržet maximálně 4 hodiny, zřídka kdy více.

Mělká práce má tedy rozhodně místo v pracovním nebo studijním rozvrhu. Jde pouze o to nedělat mělkou práci na úkor té hluboké. Zároveň by bylo ukvapené říci, že je v pořádku polovinu pracovního dne strávit v opojení mělkých činností. Spousta profesí vyžaduje časově náročné schůzku a telefonáty. Je proto klíčové nepromarnit zbylý čas a opravdu jej strávit hluboce. Také platí, že práci lze dost často odvést, i když nejsme vyloženě v nejhlubším stupni soustředění, byť lehce na úkor její unikátnosti a vytvořených hodnot.

3.6 Závěr kapitoly

Jelikož je Pomodoro technika pro řízení osobní produktivity, stručně jsem představil pojem time management. Po zasazení do kontextu a vydefinování pojmů, které se budou v této práci opakovaně používat, je již vše potřebné představeno a lze tak přejít ke konkrétním výstupům vzešlých z filosofických úvah autora techniky Pomodoro, Francesca Cirilla.

Kapitola 4

Metody time managementu v kontextu techniky Pomodoro

4.1 SMART cíle

Většina time managementových metod má společný stavební kámen a tím je právě jasné určení cílů nebo dokonce SMART cílů. Akronym sice nemá jednotné znění a různí autoři jej uvádějí s různými slovy, myšlenka však zůstává stejná. SMART je mnemotechnická pomůcka pro kvalitativní hodnocení cílů. Písmena představují[9]:

- S – Specific (Konkrétní) – Jasné a jednoznačně definované.
- M – Measurable (Měřitelný) – Existuje kritérium, podle něhož můžeme měřit plnění cíle.
- A – Achievable (Dosažitelný) – Je možné jej dosáhnout.
- R – Relevant (Relevantní) – Má smysl jej dosáhnout.
- T – Time-bound (Časově vymezený) –S harmonogramem, který má jasně časově definovaný začátek a konec.

4.1.1 Proč plánovat cíle

Plánování má jasně určit, které zdroje a aktivity nás dělí od dosažení cíle. Tím napomáhá předcházení negativních jevů jakými jsou stres nebo frustrace. Ty vždy zhoršují psychický i fyzický stav jedince a můžou zhoršovat i produktivitu práce. Pro dlouhodobé udržitelné tempo a spokojenost je tedy vhodné naučit se jim vyvarovat nebo je alespoň mírnit. Stres uvolňující techniky a strategie nám v tom pomůžou, částečně díky správnému plánování.

4.1.2 Proč je dobré odložit myšlenky na papír

Lidská mysl je schopna udržet jen omezený počet myšlenek najednou. Sestavení plánu ulehčí mysli tím, že činnosti převedeme na „papír“. Takto se můžeme zbavit myšlenek na řešení budoucích problémů a plně se soustředit na jediný úkol.Také existuje korelace mezi vytvořením plánu a splněním cílů[10].

Zavedení evidence může ze začátku sloužit jen jako vytvoření přehledu o tom, kolik pracovních intervalů připadlo daným úkolům. Po delší době vedení záznamů nám získaný přehled umožní odhadovat náročnost činností, aneb kolik úsilí si daný úkol vyžádá.

4.2 Efektivní komunikace

Dalším důvodem, proč má smysl systém Pomodoro používat je efektivnější komunikace, kdy schůzky zahrneme do plánu práce a vyhneme se tak jevu popisovaného níže.

Aktuálně žijeme v době, kdy se fenomén nazývaný instant messaging těší velké oblibě, a to nejen v soukromé sféře ale i té pracovní. Kromě Facebooku zažívá ohromující rozkvět i platforma Slack určená pro pracovní užití. Vysoce postavený produktový manažer IBM se pochlubil, že si denně pošlou 2,5 miliónu zpráv[8]. Oproštění se od plytkého vypisování a jeho nahrazení věcnými schůzkami s sebou přináší řadu výhod. Snižuje frustraci z komunikace a práce v týmu. Dále platí, že lidé, a to zejména studenti, jsou schopni ve výsledku strávit internetovou komunikací celé hodiny. Osobní schůzka však ulehčuje pochopení druhé strany a zrychluje tak celý proces. Stojí za to také zmínit sociální výhody známostí, které osobní kontakt vytváří.

4.3 Tréning hlubokou prací

Pomodoro je úzce spjato s koncentrací a koncentrace s hlubokou prací. Aplikováním Pomodora bychom se tedy měli dostávat do stavu hluboké práce. Kromě (pokud možno) kvalitních výstupů vzniklých přímo také dodatečně šlechtíme mozek tzv "vědomým tréninkem".

„Víš, jak lehký to pro mě je!?" křičí Matt Damon ve známém psychologickém dramatu Dobrý Will Hunting, než spálí matematické důkazy, protože pro jeho vrozenou genialitu byly podřadné. Postavy géniů jsou u tvůrců americké kultury velice oblíbené (další věhlasně známý příklad je Rain Man (1998)). Příslušník odvětví označovaného jako výkonností psychologie Anders Ericson však zformuloval teorii, která se staví proti americkému obrazu zrozených géniů. Jejím názvem je "vědomý trénink". Ericson napsal: "Odmítáme, že jsou tyto rozdíly mezi experty dosahujícími vynikajících výkonů a normálními dospělými pevně dané... Naopak tvrdíme, že rozdíly mezi experty dosahujícími vynikajících výkonů a normálními dospělými jsou výsledkem celoživotní snahy zlepšit svůj výkon v určité oblasti." Zvládnout kognitivně náročný úkol vyžaduje specifický typ tréninku – existuje jen málo výjimek daných vrozeným talentem[8].

Úryvek z knihy Hluboká práce[8]: *Vědomý trénink sestává ze dvou aktivit. Tou první je soustředění se na konkrétní schopnost, kterou se snažíte zlepšit, nebo na myšlenku, kterou se snažíte pochopit. Druhou aktivitou je dostávání zpětné vazby, takže můžete svůj přístup upravit, abyste zaměřovali pozornost tam, kde je to nepřínosnější. Koncentrace je stavebním kamenem*

hlubokého tréningu, jak Ericsson uvádí: "Rozptýlená pozornost je takřka opakem soustředěné pozornosti, která je pro vědomý trénink nezbytná". **Konec úryvku.**

Na Ericssonovo pozorování navázali neurovědci, kteří zjišťovali, jaké fyzické procesy stojí za progresem lidí v mentálně/znalostně náročných činnostech. V knize The Talent Code píše novinář Daniel Coyle, že příčinou zlepšování výkonu je pravděpodobně myelin. Myelin je vrstva tukové tkáně, která roste kolem nervových buněk a funguje jako izolace, díky níž buňky přenášejí vzruchy rychleji a čistěji[8]. Silnější vrstva myelinu vybudovaná tréningem z jedinců dělá kulturisty na intelektuálním poli. Stejně jako v posilovně vede izolované zapojování svalů k růstu, analogicky způsobuje intenzivní soustředění na danou schopnost to, že oligodendrocyty začnou obalovat daný obvod v mozku myelinem.

Oligodendrocyty – buňky, které obklopují neurony v centrální nervové soustavě a vytvářejí na nich myelinové pochvy[11].

Shrnutí tréningu hlubokou prací je tedy takové, že pokud izolujeme naše myšlení od vyrušení a intenzivně se soustředíme na danou činnost, zapojovaný obvod se stane silnějším díky myelinaci.

4.4 Závěr kapitoly

Kapitola představila metody time managementu, které jsou součástí fungování techniky Pomodoro. Pomodoro využívá plánování, aby její uživatel zjistil, čeho potřebuje pro splnění cíle dosáhnout a poté pomocí koncentrované práce jednotlivé úkoly splnil. Zároveň kapitola popsala pozitivní vliv opravdového soustředění na rozvoj mozku. Zmínila, že evidence úkolů je zdrojem pro zpětnou vazbu, která nám umožní naší produktivitu posunovat dále. Celek pak vytváří pracovní průběh: naplánuj, splň, vyhodnoť (a vylepši), což je široce používané a praxí ověřené workflow (pracovní styl)[12][13][14].

Kapitola 5

Srovnání technik pro řízení osobní produktivity

5.1 Pomodoro a jeho místo mezi time managementovými technikami

Podnikl jsem průzkum time managementových technik přečtením knih, webových i vědeckých článků (sekce 15) a zjistil jsem, že se zaměřují buď na dlouhodobější stanovení a organizaci úkolů nebo na splnění úkolů aktuálních, například v rámci jednoho dne. Pomodoro spadá do sekce *splnění aktuálních úkolů* a je tedy bezpředmětné srovnávat jej s technikami na způsob Bullet Journal. Bullet Journal jsem na základě přednášky nakladatelství Jan Melvil, které je vydavatelem knihy Bullet Journal, zjednodušeně definoval jako soubor všech úkolů, který slouží k jejich organizaci[3]. Smyslem Bullet Journalu je zjistit, které úkoly je třeba udělat dnes. Oproti tomu Pomodoro slouží ke splnění dnešních úkolů. Netvrdím však, že tyto dva druhy technik se vůbec nepřekrývají, neboť nemůžu začít pracovat na dnešních úkolech, pokud nevím, které to jsou. Uživatelé Pomodora, kteří nepoužívají žádnou z dlouhodobějších technik, občas Pomodoro o plánování v širším horizontu obohatí. Nejedná se však o původní zamýšlené použití techniky, a proto Pomodoro nepovažuji za konkurenta Bullet Journal a naopak.

Jako alternativu Pomodora vidím krátkodobou techniku, která nepřerušuje pracovní fázi a snaží se tak dosáhnout hluboké práce. Tuto techniku prosazuje spisovatel Cal Newport ve své knize Hluboká práce: Pravidla pro soustředěný úspěch v roztěkaném světě[8]. Je založena na dlouhých úsecích nepřerušované práce. Autor vidí smysluplnost techniky ve vysoké hodnotě výstupů, které vznikají během hlubokého soustředění. Tohoto soustředění se nejsnáze dosahuje při dlouhé nepřerušované práci. Zmiňuje však, že existují jedinci, kteří se dokážou do stavu hluboké práce ponořit okamžitě a nepotřebují tak dlouhé úseky věnované práci.

Po přečtení knihy jak zastánce Pomodora[1], Francesca Cirilla, tak zastánce hluboké práce[8], Cal Newport, jsem však nedošel k závěru, že by byla jedna technika opakem druhé. Pomodoro i Hluboká práce vyzdvihují hodnotu odpočinku pro pracovní produktivitu. Obě techniky jsou spíše odpůrcem multi taskingu a věří, že bychom naši pozornost měli upírat na jediný kognitivně

náročný úkol. Během pracovní fáze Pomodora je uživatel také odstíněn od vyrušení a je v soustředěném stavu (byť možná ne tak hlubokém, jako při dlouhých úsecích Hluboké práce).

Já vnímám rozdílnost technik především v tom, že Hluboká práce se vyskytuje zejména u lidí, kteří ve svou práci natolik věří a natolik je baví, že jsou u ní schopni vydržet hodiny a dostat se tak do hlubšího stavu pracovního nasazení. Naproti tomu Pomodoro vytváří snadnější pracovní podmínky a lze jej tak efektivně aplikovat na činnosti, které uživatel nepovažuje za zábavné a bez Pomodora by se stejně hluboce neponořil do práce. Právě tato vlastnost dle mého názoru činí z Pomodora vyhledávaného pomocníka při studiu.

Hluboká práce je spojována s velikány jako Carl Jung, kteří se snažili vytvořit originální a převratné výstupy. Pomodoro však může jednoduše využívat například student, který se musí naučit pro něj nezajímavou látku. Tím poukazuji pouze na větší flexibilitu Pomodora, netvrdím, že je určeno pro mělčí činnosti.

5.2 Přehled technik

Rozhodl jsem se uvést tyto time managementové techniky, přestože jich existuje spousta, neboť základní přístupy a metody time managementu jsou v těchto technikách obsaženy a zbylé techniky jsou spíše jejich iteracemi. Pro úplnost uvedu i techniky podobné již zmiňovanému Bullet Journalu, i když Pomodoro nenahrazují. Při sestavování přehledu jsem čerpal ze serveru Clockify[2].

Krátkodobé:

- Pomodoro.
- Timeboxing.
- Time blocking.
- The 10-Minute Rule.

Dlouhodobé:

- Bullet Journal.
- Productivity Journal.
- Kanban.
- Biological Prime Time.
- The Seinfeld Method.
- GTD - Getting Things Done.

Dále budou jednotlivé techniky stručně popsány.

5.2.1 Krátkodobé

Pomodoro

Pomodoro je založené na pravidelném střídání práce a volna. Snaží se využít času, kdy je mozek schopen intenzivně pracovat a času, který umožní mozku zregenerovat, aby se mohl vrátit k soustředěné práci.

Pro:

- Fixní pracovní čas napomáhá koncentraci.
- Pravidelné pauzy předcházejí vyhoření a zlepšují výkonnost.
- Zlepšuje odhady časové náročnosti práce.

Proti:

- Nutnost definitivně ukončit práci po uplynutí vymezeného času může být kontraproduktivní, pokud je uživatel zrovna hluboce ponořen.
- Rigidní střídání intevalů nemusí každému vyhovovat.

Timeboxing

Spočívá v alokování časových úseků (timeboxy) k aktivitám. Práce na dané aktivitě probíhá pouze během těchto daných úseků. Jakmile vyprší, práce se musí přesunout jinam nebo úplně přestat. Timeboxing většinou zahrnuje pevně dané deadliny.

Je to velice podobné Pomodoru s tím rozdílem, že u Pomodora je aktivitě většinou věnován takový počet pracovních intervalů, který si vyžádá její dokončení. Oproti tomu Timeboxing aktivitu ukončí ve chvíli, kdy se vyčerpá určený počet timeboxů (alokovaných časových intervalů). Dalším rozdílem je to, že Timeboxing nijak neklade důraz na to, aby daný box trval 25 minut. U Pomodora je většinou doporučována tato délka, ne však vynucována.

Pro:

- Perfekcionistům zabraňuje donekonečna pracovat na úkolu a neustále jej vylepšovat.
- Omezený počet časových úseků pro úkol motivuje produkovat výstupy rychleji.
- Zlepšuje odhad náročnosti.

Proti:

- Nutnost definitivně ukončit práci po uplynutí vymezeného času může být kontraproduktivní, pokud je uživatel zrovna hluboce ponořen.
- Rigidní rozvrh špatně zvládá nevyhnutelné vyrušení jako například důležité hovory.

Timeboxing lze snadno zabudovat do techniky Pomodoro a sám autor Francesco Cirillo to ve své knize doporučuje zkušenějším uživatelům techniky.

Time blocking

Probíhá tak, že si uživatel vyhradí čas pro danou aktivitu. To znamená, že do kalendáře pro aktivitu napíše počet minut, dnů, hodin a určí ji konkrétní dny a časy, kdy v tyto dny začne pracovat a kdy skončí. Zahrnuje chytré plánování a evidenci. Zároveň do určité míry definuje i konkrétní práci v daný den, proto je to dlouhodobá a krátkodobá time managementová technika. Tuto metodu proslavil Elon Musk.

Pro:

- Komplexní způsob plánování pracovního dne.
- Lze skvěle kombinovat s hlubokou prací popsanou zde 3.4.
- Navozuje pocit, že uživatel má svou práci pod kontrolou.

Proti:

- Nutnost definitivně ukončit práci po uplynutí vymezeného času může být kontraproduktivní, pokud je uživatel zrovna hluboce ponořen.
- Rigidní rozvrh špatně zvládá nevyhnutelné vyrušení jako například důležité hovory.
- Časově náročné dopředu plánovat průběh každého dne.

The 10-Minute Rule

Spočívá v deseti minutových pracovních intervalech. Uživatel si řekne, že bude pracovat po dobu deseti minut a pokud se cítí schopen pokračovat po uplynulých deseti minutách, začne další, stejně dlouhý interval. Myšlenkou je to, že je lehčí donutit se jít pracovat na deset minut, než udělat velký kus práce.

Pro:

- Zabraňuje prokrastinaci. Členění práce na takto malé intervaly nevyvolává odpor pustit se i do větších úkolů.
- Z psychologického hlediska je lehčí odpracovat 9 krát 10 minut, než jít pracovat na 90 minut.

Proti:

- Zastavovat se každých 10 minut může být dost rušivé.

5.2.2 Dlouhodobé

Biological Prime Time

Tato technika se snaží zlepšit produktivitu uživatele využíváním správného období dne, kdy jsme z biologických důvodů nejvýkonnější, pro překonání nejtěžších a nejdůležitějších úkolů.

Pro:

- Přesunování náročných úkolů na období, kdy jsme myšlenkově výkonní, a přesouvání nenáročných úkolů na tlumenější období je dobrý způsob optimalizace práce.

Proti:

- Staví na předpokladu, že jsme schopni správně určit naše produktivnější období, což se nemusí vždy povést.
- Hledání správného produktivního období znamená měnění pracovních návyků.

Spojením této techniky s Pomodorem můžeme nejtěžší činnosti přesunout v seznamu úkolů na dobu, kdy budeme mentálně nejsilnější.

The Seinfield Method

Jedná se o kalendářní systém, který je založen na myšlence :"Nepřeruš řetěz". Princip spočívá v tom, že si uživatel vytyčí aktivitu, které se chce věnovat. Konkrétně by se mohlo jednat například o cíl "Až přijdu ze školy, budu hodinu pracovat na bakalářské práci". Pokud skutečně uživatel přijde domů a úkol splní, vybarví si v kalendáři daný den. Cílem je vytvořit co nejdelší řetěz vybarvených dnů. Pokud se činnosti daný den nebude věnovat, řetěz se přeruší.

Pro:

- Pocit satisfakce z narůstajícího řetězu.
- Velice nenáročné. Pokud uživatel fyzicky nevlastní kalendář, existují jednoduché mobilní aplikace, které analogové fungování tohoto systému nahradí.

Proti:

- Občas je to nevyhnutelné a tak či tak je potřeba porušit řetěz.
- Lze zkrátka zapomenout udělat čárku do kalendáře. Zpětně je pak těžké dohledat, jestli ten den byl pořušen rětez nebo ne.

The Seinfield Method je dalším případem techniky, která může sloužit jako podpůrná technika pro udržování Pomodora.

Kanban

Tato technika staví na vizualizaci. Plocha je rozdělená pomocí sloupců na sekce odpovídající stádiu úkolu (backlog, to do, doing, done). Umožňuje přehledně sledovat, jak se úkoly pohybují napříč těmito fázemi.

Pro:

- Jednoduchá customizace (např. přidání sloupců).
- Rychlý a jednoduchý způsob znázornění průběhu projektu.

- Pocit satisfakce při přesunutí do sloupce Done.

Proti:

- Customizace může být časově náročná.
- Nijak neřadí úkoly (např. podle důležitosti).

Kanban je jednoduchá a populární technika, která může doplnit dlouhodobé plánování do Pomodora.

Komplexní dlouhodobé plánovací metody

Kanban byla prostá time managementová technika určená pro dlouhodobé plánování. Uživatelům Pomodora by mohlo vyhovovat použití některé z dalších dlouhodobých metod, které však nijak nekonkurují Pomodoru.

Jedná se o Bullet Journal, Productivity Journal, GTD – Getting Things Done. Vzhledem k jejich komplexnosti se jimi však v této práci nebudu zabývat.

5.3 Závěr kapitoly

Na poli time managementových technik stojí Pomodoro, co se popularity týče, na úplné výši[1]. Ze srovnání však vyplynulo, že techniky, které se zaměřují na splnění denních cílů, tzn. nezaměřují se na dlouhodobé plánování, jsou svým způsobem fungování velice podobné. Sám autor techniky Pomodoro ve své knížce popisuje, jak obohatit obyčejné Pomodoro o tyto další techniky a zabudovat je do něj[1].

Jediná konkurenční metoda Pomodora je dlouhá hluboká práce popisována v knížce Hluboká práce[8]. Ze srovnání těchto dvou metod však nevyplynulo, že by jedna metoda byla nadřazena druhé.

Dále se dle zadání budu zabývat pouze technikou Pomodoro.

Kapitola 6
Výzkumy podporující Pomodoro

6.1 Výzkumy

Ze sekce věnované srovnání 5.1 lze usuzovat, že rozdíly mezi technikou Pomodoro a jinými krátkodobými time managementovými technikami nejsou velké. Všechny staví na střídání práce a regenerace. Vychází ze stejných metod (smart goals atd.). Pro posuzování efektivity Pomodora budu tedy využívat i studie, které se týkají time managementu obecně.

6.1.1 Zabránění prokrastinaci

Výzkum zkoumající vliv Pomodora na prokrastinaci zjistil, že změna vnímání času, zmenšování komplexity úkolů a neagresivní metody práce jsou dobrým nástrojem pro odstranění prokrastinace[15].

6.1.2 Data z monitorující aplikace DeskTime

Studie, která sesbírala informace z aplikace DeskTime sledující produktivitu zaměstnanců, uvádí, že jimi shledaný nejefektivnější způsob práce je podobný technice Pomodoro[16]. Tato studie tvrdí, že pauzou pomůžeme nejen sami sobě ale i našemu zaměstnavateli (jinými slovy naší produktivitě).

6.1.3 Řízení času se lze naučit

Výzkum time managementu studentů ukázal výrazný nárůst schopnosti organizace času po absolvování tréningu řízení času oproti skupině, která žádný tréning nepodstoupila[17].

6.1.4 Pozitivní vliv plánování

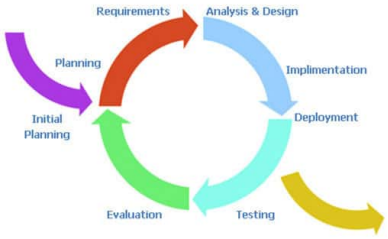
Virginia Polytechnic Institute and State University (známá také jako Virginia Tech) prezentovala data prokazující silnou korelaci mezi studentovým plánováním (organizací času) a úspěšností odevzdávání programovacích úloh[10].

6.1.5 Úspěch u agilních metodik

Technika Pomodoro má blízko k iterativnímu a inkrementálnímu vývoji softwaru, díky podobnosti svého pracovního schématu.

Pomodoro: Plánuj -> Proveď (-> Zkontroluj) -> Vyhodnoť

Inkrementální vývoj: Plánuj a analyzuj -> Implementuj -> Otestuj -> Vyhodnoť



Obrázek 6.1: Inkrementální vývoj software, zdroj: [12]

Proto je Pomodoro využíváno v agilních metodikách jako například Extreme Programming (XP), kde podle studie pomáhá lidem soustředit se na rychlost, zlepšuje retrospektivu, zmenšuje vyrušení a učí je prioritizaci. Především ale vnímají čas jako spojence[18].

6.2 Závěr kapitoly

Předešlé kapitoly vysvětlovaly, jak Pomodoro funguje a co zlepšuje, resp. proč jeho uživatele činí produktivnějšími. Předmětem této kapitoly bylo předložit výzkumy, které potvrzují fungování Pomodora a time managementu obecně, zejména pak dokládají, že má smysl se těmito metodamami pro řízeni využití času zabývat. Jelikož Pomodoro pomáhá překonávat prokrastinaci, vyrušování, zlepšuje plánování a pomáhá vnímat čas jako spojence, jedná se o dobrý nástroj pro podporu studia.

Kapitola 7

Rešerše existujících řešení

Nejúspěšnější aplikací na bázi Pomodora je Forest – Stay focused. S její rostoucí oblíbenosti se snažila celá řada dalších aplikací vydobýt si své místo na scéně, ať už s větším či menším úspěchem. Tato kapitola rozebere silné stránky již existujících řešení a také jejich nedostatky. Při sestavování rešerše jsem čerpal především z Google Play, neboť internetové články tak či tak stejně odkazovaly zpátky do tohoto obchodu. Z App Store jsem většinou nečerpal, neboť zobrazuje uživatelům méně informací, zejména skrývá počet stáhnutí aplikace (což přímo vypovídá o jejich popularitě).

Populární existující řešení:

- Forest – Stay focused
- Focus To-Do: Pomodoro Timer & To Do List
- Brain Focus Productivity Timer

7.1 Aplikace Forest – Stay focused

Forest je nejpopulárnější aplikace inspirovaná technikou Pomodoro. Dle oficiálního popisu na App Store od společnosti Apple se jedná o nejlépe hodnocenou aplikaci pro produktivitu ve 126 zemích. Má více než 2 milióny platících zákazníků. Dokonce byla doporučena v televizní reklamě “Amazing Apps“ od společnosti Apple.

Na Google Play je v kategorii aplikací s počtem stažení 10 až 50 miliónů[19](k přesným číslům veřejnost nemá přístup). Její elegantnost, jednoduchost a funkčnost proslavila time management mezi mladými lidmi.

Spočívá v zasazení stromu a určení intervalu soustředění, během kterého je uživateli zakázáno používat ostatní aplikace jeho telefonu. Pokud uživatel přeruší soustředění a použije jinou aplikaci, strom zahyne a odpočet se přeruší. Aplikaci je tak schopen okamžitě používat kdokoliv a její povedené zpracování má tendenci lidi pohltit do procesu sázení stromů a nechtějí o něj přijít. Pokud se uživatel vydržel soustředit, strom mu roste před očima a cítí tak pocit uspokojení z odvedené práce. Třešničkou na dortu je reálné zlepšování stavu přírody. Ekologie je nyní velice populární a fakt, že díky této aplikaci bylo

zasazeno 460 000 skutečných stromů (dle oficiálního popisu na App Store), jejímu jménu jenom prospívá.

Pro mě samotného byla tato aplikace vstupním bodem do světa time managementu a můžu ji doporučit každému, kdo by chtěl zkusit, zda mu vyhovuje systematický přístup k práci.

Pro:

- Jednoduchost.
- Povedený design.
- Funkčnost.
- Uživatele baví ji používat.
- Blokování rušivých aplikací.

Proti:

- Pro řadu uživatelů může působit dojmem, že má málo funkcionalit.
- Nevymezuje čas pro pauzu (některým uživatelům to však může vyhovovat).
- Na App Store je zpoplatněná.
- Neobsahuje plánování aktivit na aktuální pracovní sezení.
- Pokus o synchronizaci pc a mobilu pomocí Chrome extension skončil dle uživatelských recenzí neúspěchem[20]. Jeden uživatel zveřejnil email, kde žádal o umožnění synchronizovaného běhu, a odpověděli mu, že kvůli technickým potížím to neumožňují.

7.2 Focus To-Do: Pomodoro Timer & To Do List

Aplikace Focus To-Do se mně osobně ze všech existujících Pomodoro aplikací líbí nejvíce. Umožňuje komplexní využívání techniky. Obsahuje organizér úkolů, odhadování náročnosti a reportování. Dokonce poskytuje i částečnou multiplatformní funkcionalitu uživatelům, kteří si aplikaci koupili (prémioví uživatelé).[21].

Pro:

- Zdarma.
- Komplexní.
- Pokročilá customizace.
- Vše působí přehledně na to, kolik funkcionalit aplikace poskytuje.
- Také umožňuje "sázení stromečků".

- Blokování rušivých aplikací.
- Synchronizované úkoly.
- Premium zadarmo na 3 dny.
- Rozumná cena za cloudovou synchronizaci úkolů– 90 Kč na 3 měsíce nebo 500 Kč navždy.

Proti:

- Synchronizuje pouze úkoly, ne odpočty.
- Vadí mi polopravdivé úvodní obrázky aplikace na Google Play[21]. Na obrázku je běžící odpočet na třech zařízeních zároveň, i když přímo v popisu je uvedeno, že se synchronizují úkoly. Aplikaci jsem proto zkoušel s prémiovým účtem na dvou zařízeních zároveň a ani pomocí tlačítka "synchronize"se odpočty nepropojily. Uživatelé to v recenzích zmínili také[21].
- Neexistuje aplikace pro Linux nebo webová aplikace.

7.3 Brain Focus Productivity Timer

Velice podobná aplikaci jako Focus To-Do 7.2. Krom možnosti vypnout Wi-Fi při odpočtu však nemůžu najít důvod, proč tuto aplikaci doporučit. Je také o něco málo méně oblíbená dle uživatelských recenzí[22].

Pro:

- Zdarma.
- Poměrně komplexní.
- Customizovatelná.
- Rozumné UX.
- Placená verze obsahuje widget, přes který se aplikace příjemně ovládá.

Proti:

- Obsahuje reklamy.
- Neumožňuje "sázení stromečků".
- Není nijak multiplatformní.
- Působí méně vyladěně než Forest nebo Focus To-Do.

7.4 Výsledek srovnání

Role aplikace Forest je na trhu Pomodoro velice dominantní, byť se pravidel Pomodora až tak nedrží.

Ostatní aplikace implementují techniku Pomodoro takovým způsobem, že opravdu podporují praktikování Pomodora jak je popsáno v knížce od jejího autora[1], ne jako sázení stromečků. Možná právě díky tomu se tak neuchytily. Lidé chtějí v uvozovkách okamžité jednoduché řešení, nechtějí studovat nějakou techniku.

Všechny zmíněné aplikace 7 fungují víceméně stejně. Uživatel může zapnout, vypnout, přeskočit odpočet, někdy i vytvářet úkoly. Z nich generují reporty, někdy i multiplatformně. Neexistuje mezi nimi však použitelné multiplatformní řešení, které by umožňovalo synchronizovaný odpočet. Aplikace jsem testoval a zejména Focus To-Do: Pomodoro Timer & To Do List[21] mi byl inspirací.

7.5 Závěr kapitoly

Ze srovnání existujících řešení vyplývá, že doopravdy neexistuje jediná multiplatformní aplikace. Dokonce i zpoplatněná aplikace Forest v tomto ohledu dle uživatelů selhala. Z nespokojených uživatelských recenzí bylo zřejmé, že zájem o takovéto řešení tady je[20].

Část II

Praktická část

Kapitola 8

Analýza požadavků na aplikaci

Sběr požadavků měl za cíl určit požadované funkcionality multiplatformní aplikace na bázi techniky Pomodoro vhodné pro akademické prostředí. Co aplikace musí umožňovat a co by umožňovat mohla. Všechny požadavky především musí brát v potaz multiplatformní charakter aplikace.

8.1 Soubor požadavků

Obecné požadavky na aplikaci již byly definovány. Musí být multiplatformní, umožňovat zapínání pracovních intervalů, musí být vhodná pro akademické prostředí a jednoduchá na použití. Takováto specifikace je dobrá pro vytvoření základní představy o aplikaci, nestačí však k sestavení analýzy, která by mohla vést k vytvoření aplikace. K tomu poslouží soubor požadavků vytvořený v programu MS Excel. Hlavní výhodou je jeho jednoduchost a intuitivnost při manipulaci s tabulkami. Zároveň je díky jeho rozšířenosti nejlépe podporován a je snadné rychle nalézt řešení konkrétních problému na internetu.

Jedná se sice o komerční proprietární software, fakulta FEL však poskytuje licence jejím studentům zdarma. Typ souborů .xlsx zároveň ale umožňuje poměrně dobře funkční spolupráci s jinými produkty pro kancelářskou práci, jako například LibreOffice.

Soubor požadavků má následující jednoduchou strukturu:

- ID – ID požadavku
- Název – Název požadavku
- Popis – Popis požadavku (co, kdo, komu umožňuje dělat)

Dokument má celkem 25 požadavků a nachází se na přiloženém cd a na **gitu**. Dále jsem určil název aplikace – Felodoro. Jedná se o spojení slov FEL a Pomodoro.

8.2 Use Casy

Po vydefinování funkčních a nefunkčních požadavků na aplikaci nyní přichází čas určit, jak budou moct aplikaci využívat její uživatelé. V tom sehraje zásadní roli sestavení use casů (případů užití).

Felodoro je co se týče uživatelských práv velice jednoduchá aplikace. Pro využívání techniky Pomodoro musí být vždy uživatel přihlášen, aby mohla fungovat multiplatformní synchronizace. Přihlášení, registrace a zobrazení manuálu je dostupné i pro nepřihlášené uživatele. Není tedy třeba vytvářet Use Case diagram pro znázornění uživatelských práv. Pro definici use casů tedy opět posloužila MS Excel tabulka.

Tabulka s use casy má obvyklou strukturu:

- ID – ID use casu
- Název – Název use casu
- Aktéři – Aktér v use casu
- Cíl – Cíl use casu
- Konečný stav – V jakém stavu se bude aplikace nacházet po skončení use casu
- Vstupní podmínky – Podmínky, které musí nastat, aby mohl use case proběhnout
- Minimální záruka – V jakém stavu minimálně aplikace skončí
- Způsov vyvolání – Jak use case vyvolat
- Hlavní scénář – Ideální scénář use casu
- Alternativní scénář – Scénář, který nastane, pokud z nějaké důvodu nastala odchylka vůči hlavnímu scénáři

Dokument se nachází na přiloženém cd nebo na **gitu** a obsahuje celkem 20 use casů. Use casy vydefinovaly, jak můžou uživatelé používat aplikaci. Tato informace je stavebním kamenem pro podrobnější analýzu.

8.3 Diagram tříd

Diagram tříd definuje entity (třídy), které se v aplikaci vyskytují. Jelikož Felodoro slouží pouze k zapnutí odpočtu, vytváření úkolů a generování reportů, diagram tříd není nijak rozsáhlý. Poněvadž bývá mezi diagramem tříd a konečnou aplikací poměrně úzká vazba, jazykovou volbou pro vytvoření tohoto diagramu se stala angličtina, neboť to umožňuje jedna ku jedné mapování na databázi a kód aplikace obecně.

Diagram tříd obsahuje třídy:

- User (Uživatel) – Entita představující uživatele aplikace. Obsahuje jeho údaje pro přihlášení a customizaci.
- Task (Úkol) – Konkrétní úkol, kterému se uživatel věnuje.
- Countdown (Odpočet) – Odpočet buď pracovního intervalu nebo pauzy.

- Time (Čas) – Jednoduchý objekt složen z minut a sekund. Je využíván entitou Countdown.
- Report (Report) – Uložený report vygenerovaný na základě činností uživatele.

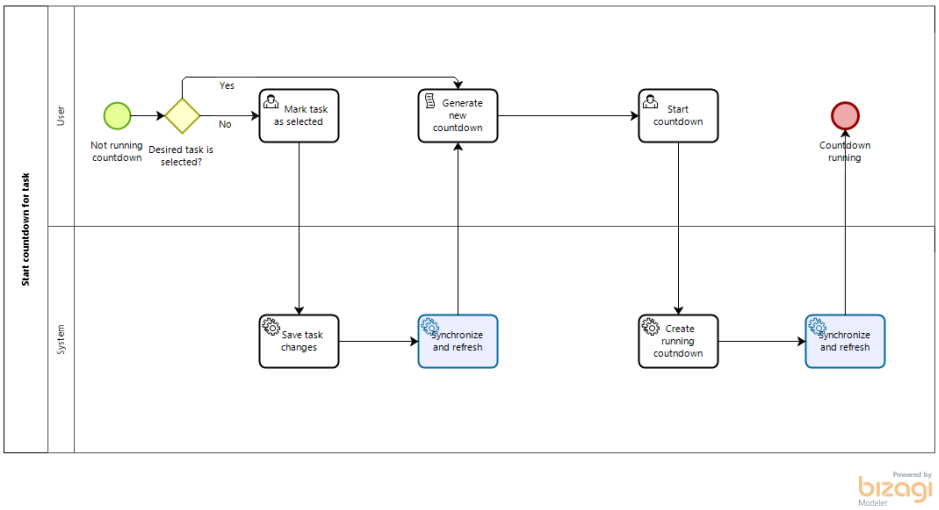
Tento diagram tříd vychází z intuitivní hierarchické struktury entit. Uživatel má úkoly. Úkoly mají své odpočty. Odpočty mají své časové údaje. A na základě údajů ze všech těchto entit jsou vytvářeny reporty.

Diagram se nachází na přiloženém cd a na **gitu**. Vychází z něj struktura databáze i objektů v kódu.

8.4 Diagramy aktivit

Diagram aktivit, většinou nazýván Process model, popisuje procesy aplikace. Pro popis procesů jsem zvolil notaci BPMN 2.0[23]. Stejně jako u diagramu tříd je použit anglický jazyk kvůli úzké vazby mezi kódem aplikace a analýzou procesů.

Nejdůležitější je proces popisující zapínání odpočtu, stavební kámen aplikace. Také je jako jediný netriviální, na rozdíl od ostatních procesů, a proto je uveden zde. Kompletní přehled diagramů aktivit se nachází na přiloženém cd a na **gitu**.



Obrázek 8.1: Proces zapni odpočet

Z diagramu vyplývá, že pokud chce uživatel zapnout odpočet, musí nejprve vybrat úkol (pokud tak už neučinil). Každý uživatel má úkol s názvem "unassigned". Pokud tedy uživatel vůbec nechce používat úkoly při praktikování metody Pomodoro, nemusí. Stačí mu jen zapínat odpočet u automaticky vybraného úkolu "unassigned".

Dalším prvek diagramu, jež stojí za povšimnutí, je modře zvýrazněný podproces synchronizace (synchronize and refresh). Jelikož je Felodoro multiplatformní aplikace, synchronizace zde hraje klíčovou roli. Po provedení jakékoliv změny klientem týkající se úkolu nebo odpočtu se změna neprojeví okamžitě v aplikaci, nýbrž proběhne synchronizace, která změnu propíše do všech aktivních zařízení klienta. Takto lze snadněji zajistit, že všichni klienti přihlášení pod stejným účtem budou mít tytéž validní data. Výjimku tvoří aktuální čas odpočtu, kde je možná až pěti sekundová odchylka (ve většině případů však rozdíl nepřekročí jednu sekundu).

8.5 Návrhy obrazovek

Návrhy obrazovek budu označovat také pojmem wireframes.

Pro vytvoření obrazovek byla použita webová aplikace Balsamiq Wireframes. Její nejsilnější stránkou je jednoduchost vytvořených obrazovek i procesu samotného vytváření. Rádoby "naškrábaný"vzhled obrazovek šetří čas tím, že nevyvolává diskuze týkající se designu, a jejich seznam předpřipravených komponent usnadňuje a urychluje skládání obrazovek.

8.5.1 Filosofie návrhu

Návrhy obrazovek jsem sestavil především v duchu jednoduchosti. Návrh tak odpovídá filosofii techniky Pomodoro, kde začínající uživatel má pouze zapínat budík[1]. Stejně tak je tomu u návrhu aplikace Felodoro. Začátečník bude pouze klikat Start u odpočtu, eventuálně Pause, pokud potřebuje udělat pauzu, a Reset, pokud hodlá dodržovat zásadu Pomodora, že přerušený odpočet se nepočítá. Tím navazují na sekundární cíl návrhu – flexibilita.

Felodoro je opensource a každý jej může upravit a používat jak chce. Tomu by mělo odpovídat i použití aplikace. Jak už bylo zmíněno, pokud chce uživatel pouze klikat jedno tlačítko, aplikace tomu je uzpůsobená. Pokud chce pracovat pouze 10 minut a odpočívat 10 minut, lze tak učinit. Pokud chce přejít o krok dál a začít používat úkoly, může využít sekci Tasks a rozšířit komplexitu aplikace. Pokud bude chtít postoupit ještě dále, je zde pro něj sekce Reports. V případě, že se chce zlepšit v odhadování náročnosti úkolů používáním techniky Pomodoro, lze přiřazovat k novým úkolům i odhad náročnosti.

8.5.2 Multiplatformní použití

Jelikož je srdcem aplikace její multiplatformní použití, zvolil jsem způsob návrhu Mobile First. Mobile First klade důraz na návrh mobilního zobrazení, neboť stále více uživatelů využívá mobil pro jejich denní elektronické činnosti[24]. Statistika návštěvnosti Facebooku z roku 2016 ukazuje, že 94 procent uživatelů přistupovalo na aplikaci pomocí mobilního zařízení a že 62 procent uživatelů k tomu používalo dokonce výhradně mobil[24].

Návrh obrazovek tedy proběhl ve stylu Mobile First. Zobrazit totéž na větším prostoru jako je počítač nebo tablet je snadné, naopak to jde hůř.

8.5.3 Shrnutí wireframů

Návrhy obrazovek zachycují celou aplikaci zobrazenou na mobilu. Vytvoření obrazovky pro počítač nebo tablet bylo vždy odvozeno z mobilního zobrazení. Na počítačích je menu nahoře ve formě tabů, neboť na to jsou uživatele zvyklí. K nahlédnutí jsou obrazovky na přiloženém cd a na **gitu**.

8.6 Závěr kapitoly

Tato kapitola se zaměřovala na požadavky a analýzu aplikace. Podkapitoly chronologicky popisují průběh specifikace zadání. Zároveň kapitola obsahuje všechny body analýzy aplikace s výjimkou diagramu nasazení. Ten totiž vychází z technologií vybraných pro implementaci, a ty jsou popsány až v následujících kapitolách. Kapitola též čtenáři nastínila, jak probíhá synchronizace a jak přibližně aplikace vypadá (úsudek na základě wireframů).

Kapitola 9

Srovnání technologií

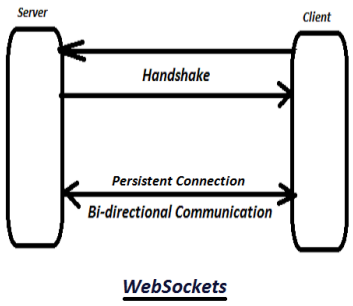
Po sestavení analýzy požadavků nastal čas určit, jak tyto požadavky nejlépe splnit. Které nástroje usnadňují implementaci přepoužíváním hotových řešení a které jsou vhodné pro potřeby aplikace co se týče zátěže a rychlosti fungování. Pomocí čeho lze vytvořit jednoduché, intuitivní a dobře vypadající uživatelské prostředí. Zároveň je při tomto rozhodování třeba zohlednit zejména multiplatformní fungování aplikace.

9.1 Technologie pro vyřešení synchronizace

Jelikož si nejvíc pozornosti zaslouží synchronizace, jako první jsou rozebrány její potřeby, zbytek technologií se odvíjí od tohoto rozhodnutí.

Synchronizace v zásadě znamená, že pokud jeden klient provede akci, sama se projeví i u druhého klienta (např. aniž by musel znovu načíst stránku). První i druhý klient v kontextu této sekce představují téhož uživatele používajícího dvě zařízení současně.

Jak zajistit, aby pokud klient jedna provede akci, server automaticky předal druhému klientovi informace o této akci? Odpovědí jsou sockety. Pro předávání informací mezi klienty se využívají web sockety. Oba klienti při otevření aplikace navážou spojení se serverem pomocí web socketu. Pokud pak jeden klient provede změnu, server může sám předat informace o změně druhému klientovi. Aplikace se pak tedy chová, jako kdyby mezi sebou klienti sami komunikovali. Princip web socketu ilustruje následující obrázek.

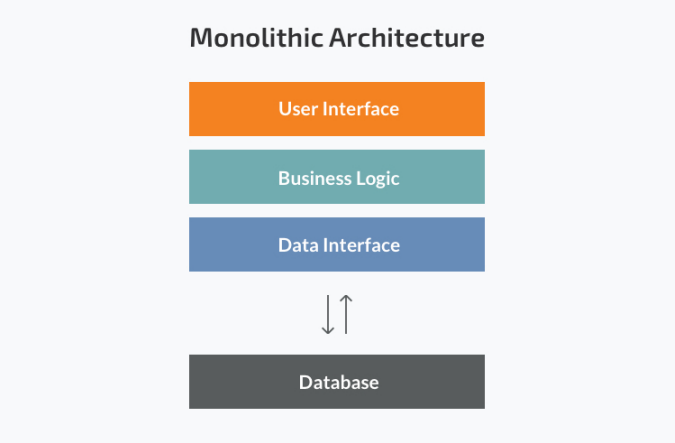


Obrázek 9.1: Web Socket – vizualizace komunikace serveru s klientem, zdroj: [25]

Pří vybírání programovacích jazyků a knihoven bylo tedy důležité, aby vše dobře spolupracovalo s web sockety.

9.2 Architektura

Co s týče architektury, není zde moc co srovnávat. Server je na bázi monolitické vícevrstvé architektury (viz obrázek 9.2), neboť je snadná pro vytvoření a nasazení. Její horší škálovatelnost oproti mikroservisám není pro rozsah aplikace Felodoro problém[26].



Obrázek 9.2: Schéma monolitické vícevrstvé architektury, zdroj: [27]

9.3 Frontend

Frontend ve schématu architektury 9.2 odpovídá User Interface.

Jelikož se jedná o Single Page Application (SPA), pro vytvoření klientské části aplikace připadá v úvahu použití některého z JavaScriptových frontendových frameworků. Nejpopulárnější frameworky jsou React.js, Vue.js a Angular[28].

9.3.1 Angular

Angular je vhodný zejména pro velice rozsáhlé aplikace a s tím je spojena i strmá učící křivka[29]. Felodoro však není žádná enterprise aplikace, a proto se Angular nezdá jako vhodná volba. Silnou stránkou Angularu býval také TypeScript. Typescript je typová nadmnožina Javacriptu a tím pomáhá předcházet celé řadě chyb[30]. Vue.js i React.js však začaly také umožňovat použití Typescriptu.

9.3.2 React.js

Oba frameworky React.js a Vue.js jsou skvělou volbou pro malou až středně velkou aplikaci, avšak React.js má příkřejší učící křivku[31]. Na druhou

stranu za ním stojí Facebook, což je pro mnoho lidí velké plus[31]. Dále se může React.js pyšnit Twittrem, Instagramem a aplikací Whatsapp[31]. Úplné prvenství zaujímá React.js v oblasti multiplatformnosti. React Native totiž umožňuje vytváření nativních Android a iOS aplikací. React tedy funguje na webu, v mobilu i v chytrých hodinkách[29].

9.3.3 Vue.js

Vue.js na sobě žádnou nálepku giganta nemá, ale i tak jej používá Gitlab, 9Gag, Nintendo a Grammarly[31]. Vue.js se chlubí jeho přístupností a mírnější učící křivkou[32]. Navíc využívá deklarativní renderování pro vytváření HTML (přímo HTML stránky), zatímco React.js využívá JSX (JavaScriptové rozšíření, které drží HTML uvnitř React elementů – JavaScript převáděný do HTML)[31]. React.js tak musí vynaložit více úsilí při provádění úkolů oproti Vue.js[31].

9.4 Backend

Backend ve schématu architektury 9.2 odpovídá Business Logic, Data Interface a Database.

Mezi programovací jazyky vhodné pro implementaci serveru vzhledem k jejich používanosti[28] a mým zkušenostem patří Java a JavaScript. Framework odpovídající Javě je pro účely srovnání Spring Boot a Express zastupuje Node.js, tedy JavaScript. Samozřejmě existuje celá řada dalších frameworků, ale tyto se těší největší popularitě[28]. Kromě toho, že se obvykle frameworky nestávají oblíbenými jen tak pro nic, to zároveň znamená, že jsou dobře podporované a existují pro ně dobré tutoriály a dokumentace, což je pro začátečníka zásadní.

9.4.1 Java - Spring Boot

Java je jedním z nejpoužívanějších programovacích jazyků, je objektově orientovaná a typová. Plusem Javy v tomto srovnání je zejména fakt, že s ní mám největší zkušenosti. Zároveň pro ní existuje skvělý framework Spring Boot. Ten vše zrychlí, zjednoduší a vytváří bezpečné aplikace[33]. Spring Boot je také nejpopulárnější Java framework na světě[34].

Databáze

PostgreSQL je open source, objektově orientována databáze. Je známá spolehlivostí, robustností a výkonem[35]. Komunikaci s databází by obstarávalo JPA – Java Persistence Api. JPA je standard programovacího jazyka Java, který umožňuje objektově relační mapování (ORM).

Spojení Spring Boot a PostgreSQL je oblíbená a pro mě dobře známá kombinace.

9.4.2 Node.js - Express

Node.js je nové řešení pro webový backend (vznikl v roce 2009) a přináší s sebou spoustu skvělých nápadů. Následující informace vycházejí z dokumentace Node.js[36].

Node.js je asynchronní JavaScriptový runtime (prostředí pro provádění scriptu). Zkrátka umožňuje exekuci JavaScriptu i mimo prohlížeč. Jedinečný je jeho přístup k práci s vlákny. Networking založený na vláknech je poměrně neefektivní a náročný na použití. Node.js byl naproti tomu navrhnut bez vláken. Je event-driven (založen na událostech) a pro eventy používá event-loop.

Tento přístup je mnohem efektivnější u neblokujících procesů než tradiční vláknování. Blokující proces je taková exekuce, která musí počkat, dokud neskončí, aby mohly probíhat ostatní procesy. Typickým příkladem je třeba náročný výpočet. Ten po celou dobu běhu vyžaduje zdroje vlákna, aby mohl skončit (a dokončit výpočet). Neblokující proces je v kontextu Node.js téměř vše ostatní. Např. dotaz na API (fetch request) stačí zavolat, ten pak čeká na odpověď (nepotřebuje zdroje vlákna), a poté je jeho odpověď zpracována.

Po dobu čekání mohlo vlákno Node.js obstarávat jiné exekuce. Až zmíněný fetch request dostane odpověď, jednoduše přidá do event loopu, že odpověď chce zpracovat. Vlákno Node.js mezitím obstarává jeden event za druhým, až se nakonec dostane ke zpracování odpovědi fetch requestu.

Aplikace Felodoro musí obstarávat právě tyto neblokující procesy. Zde přichází na řadu i dobrá spolupráce s web sockety. Při běžném vláknování se pro socket vytvoří vlákno a to obstarává komunikaci s klientem. Vlákno je tak celou dobu svázáno pouze pro tento účel. V Node.js jsou však sockety obstarávány pomocí event-loopu. Po navázáni spojení přejde vlákno na další exekuce. Až když klient bude chtít komunikovat se serverem, tak přidá event do event loopu. Event bude časem obsloužen, aniž by udržování socketu byla blokující operace.

Node.js v kombinaci s jeho minimalistickým Express frameworkem je velice populární způsob vytváření backendu[28].

Nevýhodou volby Node.js pro projekt je, že s ním nemám žádné předešlé zkušenosti (pouze s prohlížečovým JavaScriptem) a že se nehodí pro CPU náročné operace, avšak Felodoro žádné velké výpočty dělat nemusí, a proto toto omezení ničemu nevadí.

Databáze

Node.js lze stejně jako u Javy využívat spolu s PostgreSQL databázi. Při vytváření rešerše vhodných technologií jsem si však všiml, že projekty napsané v JavaScriptu se velice často pojí s databází MongoDB.

MongoDB je NoSQL open source databáze založená na dokumentech, které ukládá ve formátu BSON. Byť je oproti PostgreSQL nováček, je využívána firmami jako Facebook, ebay, Adobe, Google, SEGA, EA a SAP.

MongoDB má řadu silných stránek. Ukládání dat pomocí dokumentů je přirozenější představa než tradiční formát sloupec/řádek. BSON dokumenty

jsou opět velice přirozený způsob práce s daty, obzvlášť pro JavaScriptového programátora. BSON je totiž blízkým bratrancem formátu JSON, což je akronym pro JavaScript Object Notation. Jedná se o část JavaScriptového jazyka, která umožňuje reprezentovat objekty jako text tak, že je dobře čitelný pro člověka i stroj. JSON se stal náhradou za XML. Binary JSON (BSON) má v sobě zakódovanou délku a typ informace, což umožňuje rychlejší parsování.

Tento formát podporuje ukládání polí a vnořených dokumentů, a to umožňuje velice dynamické vytváření schématu, neboť kromě tradičních databázových vztahů pomocí referencí umožňuje i vztahy vnořením.

Integrace JSON formátu však tady nekončí. Dokonce i dotazovací jazyk je JSON.

Velkou výhodou MongoDB je, že v určitých ohledech připomíná tradiční relační databáze. Podporuje ACID transakce a joiny v dotazech.

Komunikaci s MongoDB by obstarávalo Mongoose. Mongoose je Node.js modul usnadňující přístup k objektům v databázi[37]. Funguje podobně jako ORM frameworky. Mongoose je založené na schématu, poskytuje castování, validaci, vytváření dotazů, CRUD operace a business logiku (business logic hooks)[37].

Kombinace Node.js, Express a Mongoose je často používaná a existuje pro ní řada podpůrných studijních materiálů (vzorové projekty, tutoriály, dokumentace).

9.5 API – REST vs GraphQL

Je potřeba také určit API pro komunikaci frontendu s backendem. Ve schématu architektury 9.2 to odpovídá komunikaci mezi User Interface a Business Logic.

9.5.1 REST

REST (Representational State Transfer) je architektura rozhraní pro přístup ke zdrojům pomocí HTTP volání[38].

REST je API standard[39], existuje pro něj spousta nástrojů a integrací[40]. Umožňuje také cachování[40]. Oproti GraphQL, kterému se věnuje kapitola 9.5.2, nevyžaduje sestavení schématu, což zrychluje vytváření API.

Problémy RESTu

Tradiční způsob komunikace frontendu s backendem je pomocí REST API. REST má však dva nedostatky a jsou jimi overfetching a underfetching.

Overfetching znamená, že klient musí stahovat přebytečná data. Pokud by chtěla stránka například zobrazit seznam úkolů jenom s jejich jmény a zbytek informací nezobrazovat, musí stejně stáhnout všechna data úkolů (jako jejich datum vytvoření, stav, atd.), protože endpoint /tasks by typicky poskytoval všechna data.

Druhou nevýhodou je underfetching, což znamená, že v jednom dotazu nelze získat všechny potřebné informace a musí jich tak proběhnout více.

Například pokud by stránka chtěla zobrazit aktuální úkol spolu s jeho odpočty, nestačilo by se dotázat na /tasks/id, neboť by tak klient získal pouze úkol. Pro každý jeden odpočet by musel proběhnout request na API pro získání dat daného odpočtu.

Toto šlo řešit tak, že každý API endpoint odpovídal sekci stránky a vystavoval pro ni všechna potřebná data. Takto pevné svázání backendu s frontendem však způsobovalo, že změna, ať už na backendu nebo frontendu, znamenala předělání aplikace. Frontend přestal potřebovat tolik dat? Bez předělání backendu je zde znovu overfetching. Frontend začal potřebovat více informací? Opět underfetching.

9.5.2 GraphQL

Facebook představil open source revoluci zvanou GraphQL. GraphQL je dotazovací jazyk pro API[41]. Jednoduše ho popisují následující body z oficiální GraphQL stránky[41]:

- Popiš svoje data.
- Dotaž se, na co chceš.
- Dostaň přesně to, co jsi chtěl.

Problém underfetchingu a overfetchingu tak GraphQL definitivně vyřešilo. Pokud by aplikace vyžadovala pouze názvy úkolů, může tak učnit. Pokud by aplikace vyžadovala úkoly a jejich odpočty, může všechna data získat jedním dotazem.

GraphQL je založené na schématu. To obsahuje popis datových typů, které API poskytuje, dotazy, na které je možné se zeptat a tzv. resolvery (resolvers), které určí, jak na dotazy zareagovat. GraphQL vystavuje pouze jeden endpoint (zpravidla /graphql) a nerozlišuje mezi protokoly POST a GET. Všechny dotazy jsou POST. GET request zde nahrazuje query a POST request zde nahrazuje mutation. Existuje ale i třetí typ requestu a tím je subscription.

GraphQL subscriptions

Kromě čtení pomocí query a psaní pomocí mutation aplikace často chtějí získat od serveru informace o změně dat, na kterých jim záleží. Toho lze docílit pomocí event-based subscriptions, které fungují pomocí web socketů. Klient může jednoduše odebírat změny dat, které nastaly (aniž by je sám učinil). Například na Faceboku mají odběr (subscription) na lajknutí příspěvku, který umožňuje klientovi okamžitě zjistit, že mu někdo dal palec nahoru[41].

GraphQL subscriptions jsou skvělým řešením pro synchronizaci odpočtů a úkolů v aplikaci Felodoro. Každé zařízení jednoho uživatele představuje klienta. Každý klient tak může zkrátka odebírat změny v odpočtech a úkolech a poté na ně patřičně zareagovat.

Problémy GraphQL

Oproti RESTu se jedná o poměrně novou technologii, takže je REST lépe podporovaný (integrace, cachování) a také pro něj stihlo vzniknout více návodů. Zejména jsem ale s GraphQL nikdy nepracoval a také vyžaduje sestavení schématu, což v úvodu zpomaluje vývoj API.

9.6 Výsledek srovnání

Z popisu technologií vyplynulo, že pro aplikaci je vhodnou volbou vícevrstvá architektura a web sockety.

9.6.1 Frontend

Jelikož Angular není vhodný pro malou aplikaci, rozhodování se omezilo na React.js a Vue.js. Podpora pro multiplatformnost React Native je sice velkým lákadlem, ale naučit se React a React Native a naprogramovat webovou aplikaci a poté znovu mobilní aplikaci zní jako velká časová zátěž. Proto jsem nakonec zvolil Vue.js, zejména také kvůli jeho přívětivosti vůči novým příchozím do tohoto frameworku.

9.6.2 Backend

Vybrat backend v podstatě znamenalo rozhodnout se mezi jistotou Javy nebo moderností JavaScriptu. Nakonec jsem se kvůli dobré spolupráci s web sockety rozhodl pro Node.js, spolu s Express a MongoDB. Zároveň není třeba přecházet mezi jazyky ve fullstack JavaScriptové aplikaci.

9.6.3 API

GraphQL se zdá být jako lepší volba než REST. Netrpí problémy overfetchingu a underfetchingu a má v sobě zabudovaný nástroj pro synchronizaci. Z těchto důvodu je vybráno GraphQL, byť zatím není tak zaběhlé jako REST.

9.7 Závěr kapitoly

V této kapitole byly představeny technologie vhodné pro implementaci aplikace. Po zvážení všech faktorů bylo rozhodnuto, že vhodná je monolitická vícevrstvá aplikace s backendem v Node.js (Express), databází MongoDB (Mongoose) a frontendem v JavaScriptu (Vue.js). K synchronizaci slouží web sockety v podobě GraphQL subscriptions, kde GraphQL je backendem vystavované API.

Kapitola 10
Sekundární výběr technologií

Díky kapitole 9 bylo rozhodnuto, že aplikace Felodoro je fullstack JavaScriptová aplikace. Jaké další knihovny a pluginy je ale vhodné použít pro sestavení aplikace a jak vyřešit absenci React Native pro podporu mobilních zařízení? Tato kapitola prozkoumá odpovědi na zmíněné otázky.

10.1 Podpora mobilních zařízení

Felodoro je multiplatformní, a proto má dobře fungovat i na mobilních zařízeních. Používání webového prohlížeče na mobilu je však oproti aplikacím uživatelsky velice neoblíbené. Uživatelé stráví na mobilu 90 procent času v aplikacích[42]. Programování mobilní aplikace dodatečně k webové aplikaci je však časově náročné.

10.1.1 PWA

Rozumný kompromis mobilní přívětivosti a časové náročnosti je nový pojem PWA (Progressive Web App). Tento fenomén představili v roce 2015 Frances Berriman a Google inženýr Alex Russell[43]. PWA je taková aplikace, která využívá vymoženosti moderních prohlížečů, web app manifestu a service workers pro vytvoření pseudo nativní (native-like) mobilní aplikace[43]. V praxi to vypadá tak, že si uživatel přidá skrz webovou stránku zástupce na plochu a po kliknutí na nově přidaného zástupce se stránka chová a vypadá jako opravdová aplikace. Google Chrome na Androidu dokonce sám detekuje, že se jedná o PWA a vybídne uživatele ke stáhnutí aplikace. PWA samozřejmě nedosahuje kvalit nativní aplikace, kvůli její nenáročnosti se však v některých případech jedná o dobrou alternativu. Jelikož PWA není opravdová aplikace, nezabírá také žádné místo v paměti telefonu.

Kvůli nenáročnosti a zlepšující se podpoře šlo Felodoro cestou PWA.

10.2 Nuxt.js

Na Vue.js začíná vznikat celá řada frameworků, které posunují funkcionality Vue.js dále. Jedním z nich je Nuxt.js.

Nuxt.js je progresivní Vue.js framework. Zajistí všechnu potřebnou konfiguraci pro vývoj Vue.js. Není tak třeba znovu objevovat kolo a lze snadno využívat benefity PWA nebo Google Analytics. Také automatizuje Vue.js routing. Využívá jej například Gitlab, todoist, Nespresso, Fox News channel nebo bitpay[44].

Frontend aplikace je vytvořen v Nuxt.js, neboť činí učící křivku Vue.js ještě o to mírnější. Nejedná se však o žádnou velkou změnu, Nuxt.js je v podstatě Vue.js obohacené o pár funkcionalit, jako například PWA modul nebo One Signal integrace pro snadnou implementaci push notifikací.

10.3 Vuetify

Jakmile bylo jasně určeno, že bude použit Nuxt.js a mobilní aplikace bude vyřešena formou PWA, zbývá se už jen rozhodnout, jaký styl designu aplikace zvolit. Vlastní design si vyžádá mnoho práce a jeho výsledek je u ne-grafika nejistý. Bootstrap přichází v úvahu, ale bez customizace na mně jeho komponenty působí generickým dojmem (naprosto subjektivně). Jelikož má aplikace být často používána na mobilech, rozhodl jsem se pro Material Design. Toto rozhodnutí bylo také podpořeno faktem, že mám předchozí zkušenosti s návrhem Material Designu a znám základní pokyny a pravidla návrhu.

Pro Vue.js (a tím pádem také Nuxt.js) existuje dobře podporovaná UI knihovna Vuetify, která je integrovaná do šablon Vue.js a stala se tak jasnou volbou pro aplikaci Felodoro.

10.4 Dodatečné knihovny a pluginy

Pro potřeby aplikace Felodoro jsou také použity následující knihovny a pluginy:

- graphql-compose-mongoose – Plugin, který možňuje generování GraphQL schématu, včetně resolverů a subscriptions. Automaticky vytvoří CRUD resolverů. Díky tohoto pluginu není nutné ručně psát GraphQL schéma, což jednak šetří čas a zároveň umožňuje modularizaci schématu (není celé v jednom velkém souboru).
- easytimer.js – Jednoduchá knihovna poskytující funkcionalitu časovače a odpočtu.
- axios – HTTP klient prohlížeče pro komunikaci s API.
- bcrypt – Node.js knihovna pro hashování a solení hesel.
- mobile-device-detect – Nástroj umožňující jednoduše detekovat, zda je zařízení počítač, mobil nebo tablet.
- moment.js – Knihovna pro manipulaci, parsování a validaci dat (kalendářních).
- vuelidate – Jednoduchá knihovna pro validaci Vue.js formulářů.

10.5 Závěr kapitoly

V této kapitole byly dospecifikovány poslední technologie potřebné pro vytvoření aplikace Felodoro.

Nativní mobilní aplikace je nahrazena PWA, místo čistého Vue.js je použit Nuxt.js a design aplikace obstarává Vuetify pomocí Material Designu.

Nyní už jsou známy všechny informace diagramu nasazení. Ten se nachází v příloze C.1.

Kapitola 11

Vytváření aplikace

11.1 Zdroje pro programování

Jelikož je aplikace poskládaná z technologií, s kterými jsem neměl zkušenosti, musel jsem se nejprve věnovat studiu. Krom oficiálních dokumentací Vue.js[45], Nuxt.js[44], GraphQL[41], Node.js[36] a Mongoose[37] jsem také hodně čerpal z YouTube tutoriálů kanálu TraversyMedia[46] a GitHub[47] dokumentací a komentářů uživatelů. K vytvoření backendu mi zejména pomohl článek popisující vytváření projektu, kde autor skloubil GraphQL, Node, Mongoose a Express, aniž by bylo třeba manuálně psát GraphQL schéma[48].

11.2 Vývojové prostředí

Jelikož je Felodoro fullstack JavaScriptová aplikace, rozhodl jsem se pro vytváření .js a .vue souborů použít proprietární komerční IDE WebStorm[49] od JetBrains s.r.o. Fakulta FEL poskytuje studentům licence zdarma a každý JetBrains produkt poskytuje napovídání kódu, orientaci v kódu a dobrou Git integraci.

Jako databázového klienta jsem zvolil MongoDB Compass, neboť je oficiálně doporučovaný pro MongoDB databáze[37]. Lze jej používat zdarma, placené verze obsahují support navíc. Občas jsem také používal mongo Shell[50] (zejména pro hromadné mazání).

Pří vývoji GraphQL API jsem používal opensource IDE Altair[51]. Umožňuje inteligentní napovídání pro sestavování dotazů, podporuje GraphQL subscriptions a vytvořené dotazy si lze uložit.

11.3 Prostředí pro nasazení aplikace

Jakmile byla aplikace vytvořena, bylo třeba zvolit vhodný hosting.

Nejjednodušší byla situace u databáze. Potřeby aplikace v rámci této bakalářské práce kompletně pokryl cluster zdarma od MongoDB Atlas[50]. Cluster zdarma není nijak časově omezen a poskytuje až 512MB úložiště.

U zbytku aplikace byla situace komplikovanější. Felodoro bylo vytvořeno jako jeden projekt, nemá tedy oddělený backend od frontendu, spouští se celé

najednou. To bohužel znemožnilo použití statických hostingů jako je Netlify, jelikož neumožňuje hostování standardního backendu (a už vůbec né backendu využívajícího web sockety). V úvahu tedy připadaly zdarma hostingy Now nebo Heroku. Now jsem krátce zkoušel, ale aplikaci se mi rozjet nepodařilo. Jelikož jsem už měl aplikaci nasazenou na Heroku kvůli testovacím účelům, rozhodl jsem se nic neměnit. Vycházel jsem z jednoduché úvahy: "Pokud to není rozbité, neopravuj to."Nevýhodou Heroku je, že ve verzi zdarma uspává aplikaci po půl hodině neaktivity. Zároveň se mi také nepodařilo zdarma přidat vlastní doménu, neboť se při každém pokusu objevila výzva, ať uvedu kreditní kartu.

11.4 Závěr kapitoly

Tato kapitola stručně vyobrazila vytváření aplikace, aniž by se zabývala samotným procesem psaní kódu. To by mělo pomoct čtenáři vytvořit si představu, jak vzniká aplikace na bázi Vue.js, Node.js, MongoDB a GraphQL (za použití free hostingů).

Kapitola 12

Testování

Tato kapitola se zabývá průběhem a výsledky uživatelského testování aplikace.

Aplikace Felodoro byla testována uživatelskými testy. Největší část testování jsem provedl sám, kdy jsem procházel testovací scénáře a zároveň aplikaci testoval běžným používáním. Celá tato práce byla napsaná pomocí techniky Pomodoro a celá praktická část práce pomocí aplikace Felodoro. Na Facebooku jsem také veřejně požádal zájemce o pomoc s testováním. Do aplikace se zaregistrovalo dvacet lidí, aby si ji vyzkoušeli a poskytli mi zpětnou vazbu.

12.1 Testovací scénáře

Testovací scénáře průchodem aplikací, které pokrývají základní funkcionality aplikace.

12.1.1 Scénář 1 - Kontrola přihlašování

Průchod scénářem:

1. Uživatel vyplní email, heslo a přihlásí se.

Testovací data: Existující účet uživatele. Nepřihlášený uživatel.

12.1.2 Scénář 2 - Kontrola zobrazení informativní sekce

Průchod scénářem:

1. Uživatel uvidí sekci s popisem fungování aplikace při první návštěvě.

Testovací data: Uživatel nenavštívil nikdy předtím aplikaci Felodoro (reprodukovatelné např. použitím anonymního okna prohlížeče).

12.1.3 Scénář 3 - Kontrola synchronizované manipulace odpočtu

Průchod scénářem:

1. Uživatel zapne odpočet na prvním zařízení.

2. Uživatel otevře aplikaci na druhém zařízení a uvidí běžící odpočet.
3. Uživatel pomocí druhého zařízení pozastaví odpočet.
4. Uživatel uvidí pozastavený odpočet na prvním zařízení a znovu jej spustí.

Testovací data: Uživatel je přihlášený na dvou zařízeních zároveň pod stejným účtem.

12.1.4 Scénář 4 - Kontrola synchronizované manipulace úkolu

Průchod scénářem:

1. Uživatel otevře aplikaci na dvou zařízeních.
2. Uživatel na prvním zařízení přejde do sekce Úkoly.
3. Uživatel v sekci úkoly napíše název nového úkolu a vytvoří jej.
4. Uživatel na druhém zařízení otevře detail úkolu, přejmenuje jej a potvrdí změny.
5. Uživatel na prvním zařízení uvidí provedenou změnu v sekci Úkoly a na druhém zařízení uvidí provedenou změnu na domovské stránce.

Testovací data: Uživatel je přihlášený na dvou zařízeních zároveň pod stejným účtem.

12.1.5 Scénář 5 - Kontrola zobrazení úkolů

Průchod scénářem:

1. Uživatel přejde do sekce Úkoly.
2. Uživatel uvidí stručný přehled dnešních úkolů v sekci Denní úkoly, v které se původně ocitne.
3. Uživatel zobrazí podsekci Všechny úkoly.
4. Uživatel uvidí seznam všech úkolů seskupených dle dnů. Uživatel odstraní filtr Seřadit sestupně, aby se tak mohl podívat na nejstarší úkoly.
5. Uživatel aplikuje další filtr Odložené, aby se podíval, které úkoly odložil a chtěl splnit později.

Testovací data: Přihlášený uživatel. Uživateli patří vícero úkolů. Úkoly jsou z různých dnů (včetně dnešního dne), mají odlišený stav (hotové, odložené, nové, uzavřené).

12.1.6 Scénář 6 - Kontrola zobrazení reportu

Průchod scénářem:

1. Uživatel přejde do sekce Reporty.
2. Uživatel se podívá na stručné výsledky jeho používání aplikace ve vizualizované formě.
3. Uživatel uloží report vypovídající o jeho splněných úkolech.
4. Uživatel přejde do spodní části stránky a načte uložené reporty.
5. Uživatel zobrazí detail horního reportu o splněných úkolech (odpovídá poslednímu uloženému) a prohlídne si detailnější informace o jeho používání aplikace.
6. Uživatel zobrazí graf reportu.

Testovací data: Přihlášený uživatel. Uživatel již používal aplikaci a vytvořil tak data, z kterých lze sestavit reporty.

12.1.7 Výsledky testování scénářů

Průchod scénáři jsem opakovaně testoval po dobu dvou týdnů.

Vše fungovalo s výjimkou scénáře 12.1.5. Při aplikování filtrů přestávalo fungovat seskupování úkolů podle dat jak mělo. Chyba byla opravena.

Dále jsem s narůstajícím počtem dnů testování pozoroval výrazně delší čas načítání při zapínání a pozastavování odpočtu. Díky opakovanému testování scénáře jsem tak optimalizoval načítání aktuálního odpočtu.

12.2 Uživatelské testování veřejností

Do aplikace se díky příspěvku na Facebooku zaregistrovalo dvacet lidí, kteří mi pomohli s testováním aplikace. Údaje o tomto uživatelském testování jsem získal z databáze a dodatečně jsem podrobnější informace zjišťoval dotazováním testerů přes chat.

Přes dvě třetiny uživatelů byli studenti. To je dobré, vzhledem k tomu, že Felodoro je zaměřené na akademickou půdu. Mezi uživateli byli takoví, kteří Pomodoro vůbec neznali, kteří jej používali jen jako časovač a našli se mezi nimi i pokročilejší uživatele zvyklí na využívání úkolů.

Část uživatelů byla naneštěstí spíše jen zvědavá na aplikaci, než že by si ji stáhli za účelem Felodoro testovat. Polovina uživatelů si aplikaci jen prohlédla. Maximálně zapnuli a vypnuli odpočet.

Další část uživatelů aplikaci testovala jeden den. Tři testeři pak používali Felodoro více dnů.

12.2.1 Výsledky uživatelského testování

Přijetí aplikace bylo vesměs pozitivní.

Vzhled

Všichni uživatelé až na jediného chválili design jak webu tak mobilní aplikace. Jediný nespokojený uživatel však používal starší telefon s displayem s malým rozlišením. To způsobilo, že prvky stránky byly příliš velké. Zároveň se mu samo nezobrazilo tlačítko vyvolané prohlížečem pro stáhnutí aplikace, pravděpodobně kvůli starší verzi Androidu. To zhoršilo jeho uživatelský prožitek při získáváni aplikace. Díky tohoto poznatku byl změněn návod pro získání aplikace na Androidu tak, aby i bez tlačítka bylo jasné, jak získat aplikaci.

Intuitivnost

V tomto ohledu už aplikace tolik potlesku nesklidila. Uživatelům často nebylo zprvu jasné, jak získat aplikaci, neboť jsou všichni zvyklí stahovat ji přes Google Play nebo App Store. To u PWA zkrátka není možné. Obrázkový návod s popisem je sice uveden a zvýrazněn na informativní stránce, která se uživatelům automaticky zobrazí při první návštěvě. Většina uživatelů však neměla zájem nic číst. Tohoto jsem si byl vědom a snažil jsem se udělat informativní stránku jak nejpřívětivěji to šlo. Na základě reakcí však obsahovala stále příliš textu. Jednomu uživateli se nelíbilo, že v návodu musí nejprve zvolit úroveň jeho zkušeností s technikou Pomodoro. Chtěl by mít automaticky jednu možnost předvybranou.

Jednomu uživateli také nebylo jasné, jak přeskočit pauzu. Pokud uživatelé dokončili pracovní interval a aplikaci vypnuli, tak při příštím zapnutí aplikace se jim zobrazí pauza. Pauzu je třeba totiž spustit, stopnout a až poté se zobrazí tlačítko pro přeskočení. Toto bylo v návrhu záměrně, aby se zachovala jednoduchost a nedocházelo k nevyžádanému přeskočení pauzy překliknutím.

Jeden uživatel byl nespokojen se způsobem vybírání úkolů. Čekal, že pokud vybere úkol, automaticky se mu zobrazí spuštěný odpočet u tohoto úkolu.

Funkčnost

Všichni uživatelé, kteří se vyjádřili k funkčnosti, oceňovali kvalitu multiplatformního fungování. Také se všem líbily reporty. Spokojení byli i s provedením formulářů.

Největší problém měli uživatelé s tím, že pokud zapnou odpočet, který jim běží pouze na jednom zařízení, a přejdou na jinou sekci, tak se jim odpočet resetuje. Toto je opravdu nedostatek aplikace a uživatelé jsou na něj upozorněni na informativní stránce. K mému překvapení tento problém adresovali však pouze dva uživatelé. Evidentně není přepínání mezi sekcemi během odpočtu uživateli často vyžadované. Dalo by se to vyřešit zakázáním přepnutí do jiné sekce během odpočtu. Nechtěl jsem však brát možnost používat ostatní sekce uživatelům, kteří využívají více zařízení najednou (kde tento problém není).

Uživatelům také chyběla možnost měnit sledovaný úkol během odpočtu nebo jej označit za splněný.

Nejčastěji uživatelům chyběly funkcionality, které umožňují pouze nativní aplikace. Jedná se například o blokování rušivých aplikací během odpočtu, spolupráce se Siri nebo ovládání aplikace přes widget. Na iOS aplikace nevydává žádný zvuk při skončení odpočtu, neboť Safari tuto možnost u webových stránek blokuje. A jelikož je PWA pořád jen webová stránka, mobil automaticky vypne displej při neaktivitě i během běžícího odpočtu, pokud uživatel nezmění nastavení telefonu (rychlý návod je v informativní sekci aplikace Felodoro).

12.3 Závěr kapitoly

Uživatelské testování zjistilo, že uživatelé jsou spokojení se vzhledem, multiplatformností a víceméně i intuitivností aplikace (s výjimkou sekce Úkoly), chybí jim však podpora funkcí nativních aplikací. Zejména blokování rušivých aplikací během odpočtu. Nespokojeni byli také resetováním odpočtu při přechodu na jinou sekci (aniž by jim odpočet běžel na jiném zařízení) a nemožností manipulovat s úkoly během odpočtu. Z mého osobního testování mám stejný dojem jako ostatní uživatelé.

Proces testování aplikace hodnotím kladně. Úspěšně procházejí všechny testovací scénáře. Také se zapojil poměrně velký počet uživatelů z Facebooku, kteří se podělili o jejich úhel pohledu. Jelikož byli uživatelé převážně studenti, mohli tak potvrdit, že implementace techniky Pomodoro aplikací Felodoro je dobrý pomocník při studiu.

Kapitola 13
Vyhodnocení vzniklé Aplikace

13.1 Vyhodnocení požadavků

Z celkových dvaceti pěti požadavků (dostupných k nahlédnutí na přiloženém cd nebo na **gitu**) nebyly splněny tři.

1. Požadavek na zvukové upozornění o skončení odpočtu – Nefunguje, pokud má uživatel iOS. Safari totiž blokuje automatické přehrávání zvuku[52].
2. Požadavek na Automatickou dlouhou pauzu – Nefunguje, pokud uživatel využívá přepínání mezi úkoly. Aplikace totiž zapnutí dlouhé pauzy sleduje pouze v rámci jednoho úkolu.
3. Požadavek na uživatelskou přívětivost – Sekce s úkoly dle uživatelského testování není dostatečně přívětivá.

13.2 Vzniklá aplikace

Aplikace Felodoro je první opravdu multiplatformní aplikace na bázi techniky Pomodoro. Zároveň také obsahuje komplexnější aspekty Pomodora jako vytváření úkolů a odhadování náročnosti, spolu s vytvářením reportů. Dva screenshoty z aplikace jsou k nahlédnutí v příloze D, zbytek je v adresáři projektu na gitu.

Krom webového rozhraní zároveň byla vytvořena i aplikace (pouze PWA) pro iOS a Android. Aplikace na iOS však kvůli omezením Safari funguje výrazně hůř.

Pomocí aplikace jsem vytvořil polovinu této práce a i oproti ostatním komerčním produktům jsem s Felodoro velice spokojený. Zejména mi vyhovuje možnost používat mobil jako druhou obrazovku. Typicky tak naplánuji úkoly na počítači, kde mám po ruce všechny informace pro sestavení plánu, telefon s otevřeným Felodorem zapojím do nabíječky a položím poblíž monitoru. Můžu se tak plně věnovat práci, a kdykoliv se chci podívat, kolik mi zbývá času, stačí se podívat na telefon (což odpovídá zásadě techniky Pomodoro, že odpočet by měl být vždy viditelný[1]). Během pauzy mi stačí vložit mobil do kapsy a kdykoliv a kdekoliv můžu zjistit, za jak dlouho se mám vrátit k práci.

Díky odhadování potřebných pracovních intervalů také neustále pracuji na zlepšení svých plánovacích schopností.

Toto multiplatformní fungování také otevírá dveře pro aplikování Pomodora na týmy. Pracovní týmy však většinou využívají jinou aplikaci pro delegování úkolů, a jelikož je Felodoro zaměřené na studenty, integrace s aplikacemi jako Trello nebo Evernote mu chybí. Jelikož je ale open source, kdokoliv může z Felodora vytvořit týmovou aplikaci.

13.3 Vyhodnocení implementace

Co se týče implementační stránky aplikace, tak se backend, založený na Node.js, Mongoose a GraphQL, povedl. Zejména pokud vezmu v potaz mou předešlou nezkušenost s těmito technologiemi. Stačilo pár řádků pro vytvoření CRUD GraphQL API serveru.

Na frontendu aplikace se však neblaze projevil špatný časový odhadu a nezkušenost s novými technologiemi. Na základě rychlé a uspěchané analýzy jsem se rozhodl nepoužívat Vuex (state management pro Vue.js), jelikož údajně nemusí být nutný pro menší aplikace, a také jsem se zalekl pověstné náročnosti state managementu pomocí Reduxu (React alternativa pro Vuex). To byla dle mého názoru největší chyba projektu, neboť jsem pak měl velké potíže při předávání dat mezi komponentami.

13.4 Aplikovatelnost na akademické prostředí

Technika Pomodoro je sama o sobě často vyhledávaným řešením pro překonání prokrastinace a nesoustředěnosti při studiu, například pomocí aplikace Forest[19]. Dle jednoduché implikace by pak měla být i aplikace Felodoro dobrým nástrojem pro podporu studia. Uživatelské testování však ukázalo, že to platí jen pro uživatele, kteří už techniku Pomodoro znají a rozumějí ji.

Aplikace Forest je totiž opravdu velice jednoduchá, uživatelský příjemná a využívá jen zlomek prvků techniky Pomodoro. Z tohoto důvodu je výborná pro nově příchozí, ti se totiž skrz Forest v podstatě naučí techniku Pomodoro využívat.

Felodoro je sice také jednoduchá aplikace, postrádá však výborné UX, které nové nicnetušící uživatele jasně navádí. Zároveň Forest nevymezuje čas pro pauzu, což opět působí přívětivěji na uživatelé, kteří nic neví o základech techniky Pomodoro. Forest také odrazuje od používání jiných rušivých aplikací, další velké plus pro neukázněné začátečníky.

Felodoro naproti tomu striktněji následuje pravidla užívání Pomodora[1], což mírně zvyšuje učící křivku používání aplikace. Při první návštěvě Felodoro se sice uživatelům zobrazí jednoduchý popis používání aplikace, zkušenost však ukázala, že jej nikdo číst nechce. A to je v pořádku, UX má být v ideálním případě tak dobré, aby nebylo třeba číst nic. Vytvoření takového UX je však mimo mé možnosti. Felodoro také postrádá funkcionalitu blokování jiných aplikací, jelikož tu lze nabídnout pouze pomocí nativní aplikace.

Felodoro je tedy dobrý podpůrný prostředek při studiu pomocí techniky Pomodoro, pokud však techniku neznáte, pravděpodobně Felodoro spíše vypnete, než abyste si s ním usnadnil práci.

13.5 Závěr kapitoly

Uživatelé, kteří se zúčastnili veřejného testování, hodnotili aplikaci spíše kladně. Neměli však většinou zájem používat multiplatformní funkce (byť se jim líbily), a naopak požadovali funkce, které jim může poskytnout pouze nativní aplikace. To však nebylo předmětem práce. Já osobně jsem s aplikací spokojen.

Také byly splněny téměř všechny požadavky na aplikaci. Dobré UX je zpravidla zásluhou osoby zabývající se UX, takže nepovažuji mezery Felodora v tomto ohledu za neúspěch. Nešťastná omezující politika Safari zhoršila kvalitu aplikace na iOS, ale jelikož i Facebook má problémy se Safari[53], tak toto také neberu jako velké minus.

Celkově považuji aplikaci Felodoro za úspěch a uživatelé, kteří se podíleli na testování, také měli pozitivní ohlasy. Jedná se o dobrou podpůrnou aplikaci při studiu nebo při práci, byť ji chybí odladěnost komerčních aplikací.

Kapitola 14

Další vývoj aplikace a získaná ponaučení

Uživatelské testování veřejností 12.2 zjistilo, že zájem o aplikaci Felodoro rozhodně je. Sám ji používám a vidím potenciál v zatím jediné multiplatformní Pomodoro aplikaci. Projekt bych proto chtěl dále rozvíjet.

14.1 Další naplánované kroky

Nejprve plánuji celou aplikaci přepsat do TypeScriptu. Tento projekt mi připomněl, jak důležitá je typovost. JavaScript je populárnější[54] a většina návodů a tutoriálů byla vytvořena v něm. Proto i zpětně vidím JavaScript jako dobrý začátek, i když si opravdu nedokážu představit, že bych se mu měl věnovat dlouhodobě.

Frontend předělám, aby používal Vuex. Sdílení dat napříč komponentami usnadní spolupráci jednotlivých sekcí aplikace (odpočty, úkoly) a vše pak bude působit celistvějším dojmem. Navíc si stejně jako používání JavaScriptu nedokážu představit předávání dat pomocí props(one-way data flow), kde komponenta potomka nemůže upravovat data rodiče[32].

Zároveň také nahradím Axios za Apollo[55] a pravděpodobně budu používat i Apollo server, clienta i Apollo link (socket pro graphql subscriptions)[56]. Axios určitě není nevhodný pro GraphQL fetch requesty, ale Apollo se na GraphQL zaměřuje a mají kompletní balíček GraphQL server, client, socket a dotazování/mutování. Že mi Apollo přijde jako lepší volba než Express+Axios jsem zjistil až v průběhu implementace. GraphQL dokumentace sice uvádí, že Express je doporučený způsob pro vytvoření GraphQL serveru[41], ale po tomto projektu bych všem doporučil Apollo. Je skvěle podporované, udržované, má narůstající komunitu a odhaduji, že Vue-Apollo[55] bude neoddělitelná kombinace, až se věci zaběhnou. Zejména pro pokročilejší GraphQL jako například subscriptions.

14.2 Nové funkcionality

Chtěl bych rozšířit funkcionality tak, aby umožňovaly flexibilnější používání aplikace. Z dat získaných z uživatelského testování veřejností 12.2 jsem vypozoroval, že uživatelům vyhovuje přepínání úkolů v průběhu pracovního

intervalu (byť je to částečně proti pravidlům Pomodora[1]). Obecně bych chtěl umožnit větší interakci s aplikací, zatímco běží odpočet, momentálně je to ve stylu: "udělej přípravu, zapni odpočet a pracuj". Zároveň jsem přemýšlel nad jakýmsi Easy módem aplikace, který by zlepšil přívětivost pro začátečníky.

14.3 Mobilní aplikace

Pokud by provedení těchto změn znamenalo, že aplikace by mohla mít reálnou uživatelskou základnu, začal bych s opravdovou mobilní aplikací. PWA má totiž řadu omezení, zejména pro real time aplikace jako Felodoro, a obvzlášť na iOS. Z reakcí při veřejném uživatelském testování jsem usoudil, že uživatelé nebudou mít zájem o aplikaci, která se chová hůř než ta, na kterou si už zvykli (konkrétně Forest). Proto jim tedy pravděpodobně budu muset nabídnout téměř to samé a k tomu multiplatformnost navíc (nemyslím tím však, že by Felodo mělo sázet stromy).

Tady je bohužel situace horší. Žádná plnohodnotná náhrada za React Native dle mého průzkumu zatím neexistuje, nejblíž je tomu Vue Native[57], framework ale vznikl teprve v roce 2018, nestojí za ním žádná velká společnost (pár nadšenců z Indie) a údajně se stále nejedná o framework pro produkční aplikace[58]. Už pouze fakt, že Vue Native převádí kód do React Native, je poměrně odrazující. Pokud se do doby, než dokončím práci na webové aplikaci, situace nezmění, asi budu muset přistoupit na řešení jako NativeScript-Vue[59], Ionic[60], Quasar[61] nebo PhoneGap[62].

14.4 Závěr kapitoly

Tato kapitola se zabývala budoucností aplikace. Nejprve na základě nasbíraných zkušeností s implementací dojde k úpravě kódu aplikace a dále je v plánu vytvoření opravdové mobilní aplikace. Jestli se vše podaří, toto multiplatformní řešení by na základě uživatelského průzkumu klidně mohlo najít své místo mezi aplikacemi pro řízení osobní produktivity.

Kapitola 15

Závěr

V rámci práce byla provedena analýza pojmů a metod, které se týkají řízení osobní produktivity a časového plánování. Byly zmíněny i další time managementové techniky, předmětem práce byla však technika Pomodoro. Dále byla vytvořena rešerše existujících aplikací na bázi metody Pomodoro.

Analýza time managementových metod a rešerše aplikací posloužily jako základ pro návrh a vytvoření nové aplikace, která umožňuje jednoduché a multiplatformní využití techniky Pomodoro v akademickém prostředí. Po nasazení aplikace proběhlo uživatelské testování, do kterého se zapojilo dvacet lidí. Výstupy testování posloužily k vytyčení dalšího směřování aplikace.

Byly tedy naplněny všechny cíle práce (analýza metod time managementu, analýza a návrh aplikace, implemenetování a otestování aplikace).

Osobně pro mě byl projekt výzvou, neboť synchronizovaný multiplatformní odpočet Pomodoro aplikace zatím nikým implementován nebyl. Nepovedlo se to ani nejúspěšnější Pomodoro aplikaci Forest[20]. Cíle projektu se však podařilo naplnit.

Synchronizace však funguje a uživatelské přijetí bylo také poměrně pozitivní. Webový klient a Android aplikace fungují správně, iOS je kvůli omezování Safari horší.

Práce s názvem Aplikace pro podporu metody Pomodoro byla kompletně napsaná za pomocí techniky Pomodoro a z poloviny přímo pomocí vzniklé aplikace Felodoro.

Projekt považuji za úspěch a kromě vzniklé aplikace pro mě je přínosem také nabytá zkušenost s MongoDB, Node.js, GraphQL a Vue.js.

Jinak je práce přínosem zejména pro ty, kteří chtějí využívat metodu Pomodoro na více zařízeních. Aplikace Felodoro jim umožní nejen to, ale i možnost využívat úkoly a reporty pro komplexnější uživatele Pomodora.

Zároveň je Felodoro open source a obsahuje jednoduchý návod na lokální spuštění aplikace. Kdokoliv si tak může aplikaci upravit nebo se inspirovat způsobem, jakým Felodoro kombinuje MongoDB, Node.js, GraphQL a Vue.js.

Přílohy

Příloha A

Literatura a zdroje

1. CIRILLO, Francesco. *Technika Pomodoro: legendární systém pro plánování času a dokonalou koncetraci během hluboké práce*. Vydání první. V Brně: Jan Melvil Publishing, 2019. ISBN 978-80-7555-069-9.
2. *Time management techniques* [online]. San Francisco: Clockify, 2017 [cit. 2020-01-06]. Dostupné z: `https://clockify.me/time-management-techniques`.
3. *Přednáška o Pomodoru* [online]. Česká Republika: Melvil, 2019 [cit. 2020-01-06]. Dostupné z: `https://www.youtube.com/watch?v=f2W3RuH4TGc`.
4. *A guide to the project management body of knowledge (PMBOK guide)*. 3rd ed. Newtown Square, Pa.: Project Management Institute, Inc., c2004. ISBN 1930699506.
5. COVEY, Stephen R. *7 návyků vůdčích osobností pro úspěšný a harmonický život: návrat etiky charakteru*. 1. vyd. Praha: Pragma, 1994. ISBN 80-85213-41-9.
6. LUDWIG, Petr. *Konec prokrastinace: [jak přestat odkládat a začít žít naplno]*. Vyd. 1. V Brně: Jan Melvil, 2013. ISBN 978-80-87270-51-6.
7. *Online Etymology Dictionary* [online]. Pennsylvania: Etymonline, 2007 [cit. 2020-01-06]. Dostupné z: `https://www.etymonline.com/word/procrastination`.
8. NEWPORT, Cal. *Hluboká práce: pravidla pro soustředěný úspěch v roztěkaném světě*. Vydání první. V Brně: Jan Melvil Publishing, 2016. ISBN 978-80-7555-008-8.
9. *Time management* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2020-01-06]. Dostupné z: `https://en.wikipedia.org/wiki/Time_management`.
10. RÖSSLING, Guido. *Correlation between time management and successful outcome - Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* [online]. New York, NY: ACM, 2011 [cit.]. ISBN 9781450306973. Dostupné z: `http://delivery.acm.org/10.1145/2000000/1999842/p331-shaffer.pdf?ip=147.32.31.197&id=1999842&acc=ACTIVE%20SERVICE&key=`

D6C3EEB3AD96C931%2E9BD1EC80ACA8C1C5%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1568474109_347497642ab216bbcaead88d74025f0e#URLTOKEN.

11. DVOŘÁK, Karel. *Atlas patologie pro studenty medicíny: Oligodendroglie: Masarykova univerzita - Atlas patologie pro studenty medicíny: Oligodendroglie* [online]. Brno: MUNI, 1990 [cit. 2020-01-06]. Dostupné z: https://atlases.muni.cz/atlases/stud/atl_cz/main+cnspatol+reakcecns.html%5C#reakceneugli+oligodnd.

12. *Incremental software development schema* [online]. Indie: Testing Excellence, 2018 [cit. 2020-01-07]. Dostupné z: https://www.testingexcellence.com/incremental-model/.

13. STACK, Laura. *Theproductivitypro: the-work-less-more-success-guide-to-time-management-step-five* [online]. Colorado: Theproductivitypro, 2012 [cit. 2020-04-21]. Dostupné z: https://theproductivitypro.com/blog/2012/07/the-work-less-more-success-guide-to-time-management-step-five/.

14. *Visual-paradigm: extreme-programming-vs-scrum* [online]. Hong Kong: Visual Paradigm International Ltd., 2018 [cit. 2020-04-21]. Dostupné z: https://www.visual-paradigm.com/tw/scrum/extreme-programming-vs-scrum/.

15. FENG, Jia. An evaluation of the Pomodoro Technique for stopping procrastination and behaviour change. In: 2016.

16. *Secret of the Most Productive People - Breaking | DeskTime Insights* [online]. 2018 [cit. 2019-09-14]. Dostupné z: https://desktime.com/blog/17-52-ratio-most-productive-people.

17. NADINLOYI, Karim Babayi; HAJLOO, Nader; GARAMALEKI, Nasser Sobhi; SADEGHI, Hasan. The Study Efficacy of Time Management Training on Increase Academic Time Management of Students. *Procedia - Social and Behavioral Sciences* [online]. 2013, roč. 84, s. 134–138 [cit. 2019-09-14]. ISSN 18770428. Dostupné z DOI: 10.1016/j.sbspro.2013.06.523. PII: S1877042813015905.

18. GOBBO, Federico; VACCARI, Matteo. The Pomodoro Technique for Sustainable Pace in Extreme Programming Teams. In: 2008. Dostupné z DOI: 10.1007/978-3-540-68255-4_18.

19. *Forest aplikace - Google Play* [online]. Google Play: Google, 2020 [cit. 2020-04-29]. Dostupné z: https://play.google.com/store/apps/details?id=cc.forestapp%5C&hl=en_US.

20. *Forest Extension store* [online]. USA: Google, 2019 [cit. 2020-01-07]. Dostupné z: https://chrome.google.com/webstore/detail/forest-stay-focused-be-pr/kjacjjdnoddnpbbcjilcajfhhbdhkpgk?hl=cs.

21. *Focus To-Do: Pomodoro Stopky & Úkolníček* [online]. Google Play: Focustodo@163.com, 2020 [cit. 2020-04-29]. Dostupné z: https://play.google.com/store/apps/details?id=com.superelement.pomodoro.

22. *Brain Focus Productivity Timer* [online]. Google Play: Brainfocus, 2020 [cit. 2020-04-29]. Dostupné z: `https://play.google.com/store/apps/details?id=com.AT.PomodoroTimer`.
23. *BPMN 2 specifikace* [online]. USA: Object Management Group, Inc., 2014 [cit. 2020-05-03]. Dostupné z: `http://www.bpmn.org`.
24. *Mobile First: Co je to "Mobile First"? Ale doopravdy* [online]. Česká Republika: Martin Michálek, 2015 [cit. 2020-04-23]. Dostupné z: `https://www.vzhurudolu.cz/prirucka/mobile-first`.
25. *Web Socket Image* [online]. USA: Infoworld, 2016 [cit. 2020-04-25]. Dostupné z: `https://www.infoworld.com/article/3088338/how-to-work-with-web-sockets-in-net.html`.
26. *Monolitická architektura vs Microservices* [online]. USA: Medium, 2015 [cit. 2020-04-25]. Dostupné z: `https://articles.microservices.com/monolithic-vs-microservices-architecture-5c4848858f59`.
27. *Monolitická architektura* [online]. USA: Experfy, 2018 [cit. 2020-04-25]. Dostupné z: `https://www.experfy.com/blog/microservices-vs-monolith-which-architecture-is-the-best-choice-for-your-business`.
28. *Stack Overflow Survey* [online]. USA: Stack Exchange, Inc., 2019 [cit. 2020-04-25]. Dostupné z: `https://insights.stackoverflow.com/survey/2019`.
29. *Angular* [online]. USA: Medium, 2018 [cit. 2020-04-25]. Dostupné z: `https://blog.usejournal.com/3-reasons-why-angular-6-is-the-future-of-enterprise-scale-web-applications-7bac6edf133`.
30. *Angular TypeScript* [online]. Medium: Nrwl.io, 2016 [cit. 2020-05-03]. Dostupné z: `https://vsavkin.com/writing-angular-2-in-typescript-1fa77c78d8e8`.
31. *VueJS vs ReactJS* [online]. Canada: Towards Data Science Inc., 2020 [cit. 2020-04-25]. Dostupné z: `https://towardsdatascience.com/vuejs-vs-reactjs-which-will-reign-in-2020-c5591213784c`.
32. *VueJS* [online]. Čína: Evan You, 2019 [cit. 2020-04-25]. Dostupné z: `https://vuejs.org`.
33. *Spring Boot* [online]. USA: VMware, Inc., 2020 [cit. 2020-04-25]. Dostupné z: `https://spring.io/projects/spring-boot`.
34. *JVM Ecosystem report* [online]. UK: Snyk Ltd., 2018 [cit. 2020-05-06]. Dostupné z: `https://snyk.io/blog/jvm-ecosystem-report-2018-platform-application/`.
35. *PostgreSQL* [online]. USA: The PostgreSQL Global Development Group, 2020 [cit. 2020-04-25]. Dostupné z: `https://www.postgresql.org`.
36. *Node.js* [online]. USA: Joyent, Inc., 2014 [cit. 2020-04-25]. Dostupné z: `https://nodejs.org/en/`.

37. *Mongoose* [online]. GitHub: Mongoosejs, 2018 [cit. 2020-04-25]. Dostupné z: https://mongoosejs.com/.

38. *REST* [online]. USA: World Wide Web Consortium., 2004 [cit. 2020-05-03]. Dostupné z: https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/%5C#relwwwrest.

39. *REST API popularita* [online]. Izrael: RestCase, 2015 [cit. 2020-05-06]. Dostupné z: https://blog.restcase.com/the-rise-of-rest-api/.

40. *GraphQL vs REST* [online]. USA: Meteor Development Group Inc., 2017 [cit. 2020-04-25]. Dostupné z: https://www.apollographql.com/blog/graphql-vs-rest-5d425123e34b.

41. *GraphQL* [online]. USA: The GraphQL Foundation, 2020 [cit. 2020-04-25]. Dostupné z: https://graphql.org.

42. *Web vs aplikace* [online]. USA: JMango360, 2018 [cit. 2020-04-26]. Dostupné z: https://jmango360.com/wiki-pages-trends/mobile-app-vs-mobile-website-statistics/.

43. *PWA* [online]. USA: Infrequently, 2015 [cit. 2020-04-26]. Dostupné z: https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/.

44. *Nuxt.js* [online]. Amsterdam: Nuxt.js, 2019 [cit. 2020-04-26]. Dostupné z: https://nuxtjs.org.

45. *Vuetify* [online]. USA: Vuetify, LLC, 2016 [cit. 2020-04-25]. Dostupné z: https://vuetifyjs.com/en/.

46. *Traversy Media YouTube* [online]. YouTube: YouTube, 2019 [cit. 2020-04-27]. Dostupné z: https://www.youtube.com/channel/UC29ju8bIPH5as8OGnQzwJyA.

47. *GitHub* [online]. GitHub: GitHub, 2019 [cit. 2020-04-27]. Dostupné z: https://github.com.

48. *GraphQL API s Node, Mongoose a Express* [online]. USA: Stream.io, Inc., 2019 [cit. 2020-04-27]. Dostupné z: https://getstream.io/blog/tutorial-create-a-graphql-api-with-node-mongoose-and-express/.

49. *WebStorm* [online]. Česká Republika: JetBrains s.r.o., 2020 [cit. 2020-04-27]. Dostupné z: https://www.jetbrains.com/webstorm/.

50. *MongoDB* [online]. New York: MongoDB, Inc., 2020 [cit. 2020-04-25]. Dostupné z: https://www.mongodb.com.

51. *Altair GraphQL* [online]. USA: Altair, 2019 [cit. 2020-04-27]. Dostupné z: https://altair.sirmuel.design.

52. *HTML5 audio limitations* [online]. USA: IBM, 2012 [cit. 2020-04-28]. Dostupné z: https://www.ibm.com/developerworks/library/wa-ioshtml5/wa-ioshtml5-pdf.pdf.

53. *Facebook - podporované prohlížeče* [online]. USA: Facebook, 2020 [cit. 2020-04-28]. Dostupné z: https://www.facebook.com/help/211644178877843.

54. *TypeScript vs JavaScript* [online]. USA: ITNEXT, 2019 [cit. 2020-04-29]. Dostupné z: https://itnext.io/choosing-typescript-vs-javascript-technology-popularity-ea978afd6b5f.

55. *Vue Apollo* [online]. Git Hub: Vue Apollo, 2018 [cit. 2020-04-29]. Dostupné z: https://apollo.vuejs.org.

56. *Apollo GraphQL dokumentace* [online]. GitHub: Apolloh GraphQL, 2020 [cit. 2020-04-29]. Dostupné z: www.apollographql.com/.

57. *Vue Native* [online]. Indie: GeekyAnts, 2018 [cit. 2020-04-29]. Dostupné z: https://vue-native.io.

58. *Vue Native kritika* [online]. Indie: Medium, 2019 [cit. 2020-04-29]. Dostupné z: https://medium.com/datadriveninvestor/vue-native-is-best-to-create-native-apps-here-is-why-d018d96dd8e9.

59. *NativeScript-Vue* [online]. USA: Progress, 2019 [cit. 2020-04-29]. Dostupné z: https://nativescript-vue.org.

60. *Ionic* [online]. USA: Drifty, 2013 [cit. 2020-04-29]. Dostupné z: https://ionicframework.com.

61. *Quasar* [online]. GitHub: Quasar, 2019 [cit. 2020-04-29]. Dostupné z: https://quasar.dev.

62. *Adobe PhoneGap* [online]. USA: Adobe, 2009 [cit. 2020-04-29]. Dostupné z: https://phonegap.com.

Příloha B

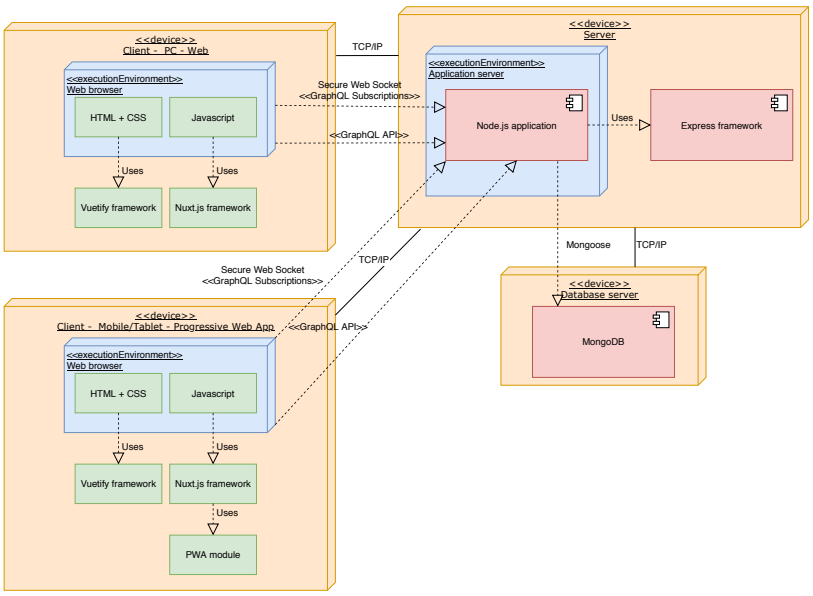
Odkaz adresáře projektu na školní GitLab

- Krátký odkaz: https://bit.ly/2SUBClV
- Celý odkaz: https://gitlab.fel.cvut.cz/lipowada/pomodoro-method-application

Obrázek B.1: QR kód odkaz

Příloha C

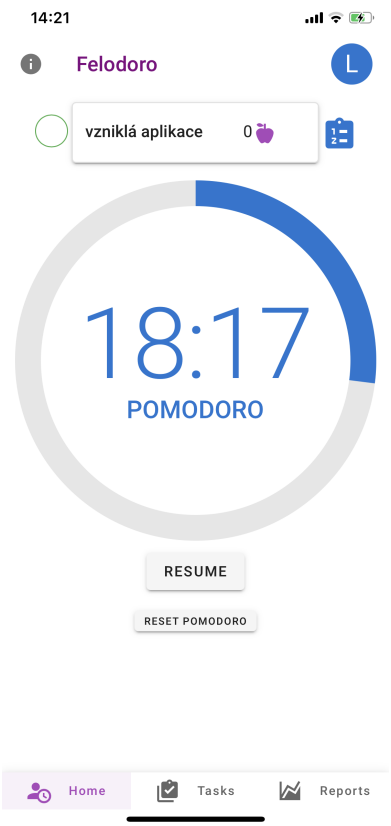
Diagram nasazení



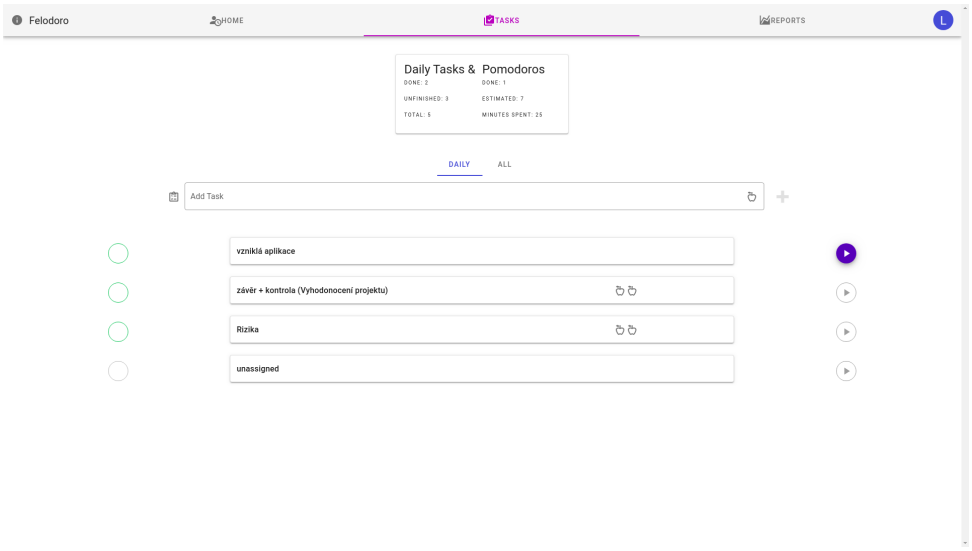
Obrázek C.1: Diagram nasazení

Příloha D

Screenshoty z aplikace



Obrázek D.1: Domovská stránka v mobilní aplikaci



Obrázek D.2: Sekce úkolů