Smjernice za izradu dokumentacije

Programsko inženjerstvo kao i druge inženjerske discipline zahtijeva vođenje dokumentacije o provedenim aktivnostima, donesenim odlukama, arhitekturi, strukturi i dizajnu programskog proizvoda, očekivanom ponašanju, rezultatima testova i slično. Stoga ste tijekom provedbe projekta razvoja dužni dokumentirati svaku provedenu aktivnost. U nastavku (vidi Tablica 1) opisujemo okvirnu strukturu dokumentacije koju biste trebali izraditi u sklopu vašeg timskog projekta. Napominjemo da cjelokupna dokumentacija treba biti izrađena na Wiki sustavu vašeg timskog GitHub repozitorija.

Tijekom rada na vašim projektima uzmite u obzir da bi vaš cjelokupni pristup i rad trebao biti orijentiran na provedbu odabrane metodike razvoja, to jest provedbu svih faza razvoja programskog proizvoda. Iako u praksi članovi tima najčešće dobivaju izolirane, manje, tematske zadatke i odgovornosti, mi smo dogovorili da će studenti tijekom rada na projektima imati tzv. vertikalne odgovornosti. To znači da bi svaki član tima, za vlastite funkcionalnosti trebao provesti cjelokupni proces razvoja po odabranoj slijednoj metodici ("od specifikacije zahtjeva do isporuke i dokumentiranja"). To znači da se, između ostalog, od svakog člana tima očekuje da ravnopravno sudjeluje u zajedničkim dijelovima dokumentacije, te da samostalno dokumentira funkcionalnosti za koje je zadužen.

Tablica 1 Struktura dokumentacije

Br.	Poglavlje	Upute
1.	O projektu	Ukratko opisati temu projekta.
2.	Projektni tim	Navesti osnovne podatke o članovima tima, te dogovorena zaduženja (u skladu s prihvaćenom prijavom i sugestijama mentora).
3.	Plan projekta	Navesti odabrani metodološki pristup, kreirati terminski plan projekta, gantogram, okvirne troškove, te ponudu za fiktivnog naručitelja. Okvirno, potpoglavlja bi mogla biti: 3.1. Metodološki pristup 3.2. Terminski plan projekta 3.3. Zaduženja članova tima 3.4. Izračun troškova 3.5. Ponuda naručitelju
4.	Izvješća o provedbi	Bilo bi dobro tijekom provedbe projekta povremeno analizirati ostvarenost po ključnim metrikama, te temeljem rezultata analize napraviti dorade i korekcije projektnog plana.
5.	Specifikacija korisničkih/softverskih zahtjeva. U ovom poglavlju se može struktura preuzeti iz IEEE standarda ili drugih predložaka za specifikaciju zahtjeva.	
5.1.	Uvod	
5.1.1.	Svrha	Opišite svrhu SRS dokumenta (ne softverskog rješenja). Navedite kome je namijenjen dokument, tj. tko treba dokument čitati i razumjeti.
5.1.2.	Opseg	Ukratko objasnite problemsku domenu kojom se rješenje bavi. U kojem kontekstu će se softversko rješenje upotrebljavati? Dajte

		ime/naziv softverskom rješenju kojim će ga se u ostatku dokumenta moći naslovljavati, te po potrebi verziju. Objasnite radi li se o potpuno novom rješenju ili nadogradnji. Navedite što će softver raditi i što neće raditi. Koje dobrobiti i unaprjeđenja očekujemo da će softversko rješenje donijeti.
5.1.3.	Definicije, akronimi, skraćenice	Definirajte osnovne pojmove, akronime i skraćenice koji se koriste u specifikaciji, a koje su potrebne da bi se moglo ispravno razumjeti SRS dokument.
5.1.4.	Reference	Referencirajte sve dokumente, web stranice, standarde, druge specifikacije koji se spominju u SRS-u.
5.1.5.	Struktura dokumenta	Ukratko opišite kako je ostatak dokumenta organiziran i što sadrži.
5.2.	Općeniti opis	
5.2.1.	Perspektiva proizvoda	Stavite softversko rješenje u kontekst i odnos s drugim povezanim sustavima. Navedite je li softversko rješenje neovisno i u potpunosti samostalno, ili predstavlja dio većeg sustava (što je čest slučaj), ili je zamjena za neki postojeći sustav. Ako je dio većeg sustava potrebno je staviti u odnos zahtjeve cjelokupnog sustava sa zahtjevima našeg softverskog rješenja, te jasno definirati sučelja između njih. Osim odnosa s eventualnim nadređenim sustavom, navesti i vanjske sustavima i softver koji vaše softversko rješenje koristi. To mogu biti npr.: DMBS, operacijski sustav, web servisi i API-ji. U slučaju da softversko rješenje izravno koristi hardver i komunikacijske tehnologije opišite i značajke sučelja s njima (vrsta hardvera, port, komunikacijski protokoli, bluetooth, NFC, IC, TCP, format razmjene podataka i sl.).
5.2.2.	Funkcije proizvoda	Navedite glavne funkcije koje softversko rješenje treba sadržavati, bez ulaženja u detalje svake od tih funkcija (detaljni zahtjevi će biti sadržani u poglavlju 3). Funkcije trebaju biti organizirane i opisane na način da već pri prvom čitanju budu razumljive svim čitateljima dokumenta, uključujući i naručitelja/korisnika. Za formulaciju funkcija softverskog rješenja na ovoj razini može poslužiti specifikacija više razine, npr. specifikacija korisničkih zahtjeva (ako postoji).
5.2.3.	Karakteristike korisnika	Identificirajte grupe/uloge korisnika za koje očekujemo da će koristiti softversko rješenje (npr. nastavnik, profesor, administrator, blagajnica, bankovni službenik, računovođa, …). Navedite opće karakteristike identificiranih korisnika, navedite po čemu se oni razlikuju. To može uključivati razinu obrazovanja, iskustva, računalne i tehničke pismenosti, zatim predviđenu učestalost korištenja softverskog rješenja, razinu dodijeljenih dozvola, skupa funkcija softvera kojeg će koristiti.
5.2.4.	Ograničenja	Navesti aspekte problemske domene i samog rješenja koji mogu djelovati ograničavajuće na razvoj softverskog rješenja. To može uključivati: zakonske i korporativne propise i regulative (npr. GDPR odredbe, fiskalizacija, sigurnosne politike, i sl.); hardverska ograničenja (npr. u razvoju za male uređaje – baterija, memorija, procesor, povezivost, i sl.); potrebu za prilagodbom drugim sustavima (npr. interakcija s postojećim i potencijalno zastarjelim sustavima); sigurnosnu kritičnost aplikacije (npr. u slučaju da aplikacija upravlja opasnim strojevima, medicinskim uređajima, ili radi sa osjetljivim i povjerljivim podacima, i sl.); zahtjeve za pouzdanošću (npr. može biti

<table>
<tr><td></td><td></td><td>zahtijevano korištenje određenog pristupa, alata praksi i standarda u programiranju, modeliranju i testiranju softverskog rješenja i sl.)</td></tr>
<tr><td>5.2.5.</td><td>Pretpostavke i ovisnosti</td><td>Navedite pretpostavke i otvorena pitanja (za razliku od poznatih činjenica) čiji naknadni ishod može utjecati na zahtjeve navedene u ovom dokumentu. To su vrlo često okolnosti koje nisu pod našom odgovornosti (npr. ako postoji vjerojatna mogućnost ili najava promjene zakonske regulative koja će rezultirati modificiranjem definiranih zahtjeva; ili izlazak nove verzije API-ja o kojem naš softver ovisi).</td></tr>
<tr><td>5.2.6.</td><td>Ostalo</td><td>Navedite ostale aspekte problemske domene i budućeg softverskog rješenja koje smatrate bitnima a koji nisu predviđeni u ostalim dijelovima dokumenta.</td></tr>
<tr><td>5.3.</td><td>Funkcionalni zahtjevi</td><td>Definirajte funkcionalne zahtjeve za softversko rješenja i to na način da pružite dovoljno informacija dizajnerima i programerima da mogu započeti sa osmišljavanjem i implementacijom rješenja, a testerima da osmisle testne slučajeve. Zahtjevi navedeni ovdje se temelje na funkcijama proizvoda opisanim u poglavlju 5.2.2., ali su opisani s višom razinom detalja. Fokus je na funkciji i ograničenjima sustava. Svakom zahtjevu treba dodijeliti jedinstven identifikator. Zahtjeve je moguće i grupirati po različitim kriterijima, kao npr. korisnicima (nastavnik, student, blagajnik, administrator…), slučaju korištenja, domenskim konceptima, i sl.

Za svaki zahtjev treba popuniti sljedeću tablicu:
<table>
<tr><td>Identifikator</td><td>Jedinstveni identifikator zahtjeva.</td></tr>
<tr><td>Zahtjev</td><td>Opis zahtjeva u obliku „Sustav će omogućiti <funkcionalnost> <objekt> uz <ograničenja>“. Stil pisanja treba biti ujednačen. Prilikom formulacije vodite se prihvaćenim smjernicama za definiranje zahtjeva, kao npr. onima navedenim u INCOSE Smjernicama za pisanje zahtjeva.</td></tr>
<tr><td>Obrazloženje</td><td>Obrazloženje zašto zahtjev postoji/zašto je potreban.</td></tr>
<tr><td>Način provjere</td><td>Kriterij provjere ili testni scenarij koji će omogućiti utvrđivanje je li zahtjev ispunjen ili nije.</td></tr>
<tr><td>Prioritet [1-5]</td><td>Prioritet zahtjeva (1 – najveći prioritet, 5 najmanji prioritet)</td></tr>
<tr><td>Izvor/Porijeklo</td><td>Naziv dokumenta kojim je zahtjev propisan ili dionika koji je podnio zahtjev.</td></tr>
</table>
</td></tr>
<tr><td>5.4.</td><td>Nefunkcionalni zahtjevi</td><td>Navesti nekoliko nefunkcionalnih zahtjeva koji bi bili od izrazitog značaja za softverski sustav. To npr. mogu biti sljedeći nefunkcionalni zahtjevi:</td></tr>
<tr><td>5.4.1.</td><td>Izgled softvera</td><td>Navedite zahtjeve (ukoliko postoje) koji su povezani s izgledom i vizualnim stilom softvera. To može uključiti zahtjeve da se npr. koristi formalan i korporativan stil (u slučaju poslovne aplikacije), ili zaigran stil (npr. računalna igra za djecu); zahtjev za korištenjem konkretne palete boja ili kontrola kako bi aplikacija bila u skladu s brendiranjem poduzeća i sl.</td></tr>
<tr><td>5.4.2.</td><td>Upotrebljivost softvera</td><td>Navedite zahtjeve (ukoliko postoje) koji su povezani s lakoćom učenja i korištenja softvera, prilagodbom za osobe s poteškoćama, lokalizacijom. To može uključiti zahtjeve vezane uz krivulju učenja, brzinu korištenja aplikacije (npr. brzina unosa podataka), lakoću pamćenja opcija softvera, frekvenciju grešaka koje korisnik napravi u</td></tr>
</table>

		radu sa softverom, mogućnost odabira jezika, mogućnost korištenja od strane slijepih osoba i sl.
5.4.3.	Performanse softvera	Navedite zahtjeve (ukoliko postoje) koji su povezani s performansama softvera. To može uključiti zahtjeve vezane uz brzinu procesiranja zadataka, odziv, preciznost rezultata, kapacitet pohrane, skalabilnost, dostupnost sustava i sl.
5.4.4.	Izvođenje softvera i okruženje	Navedite zahtjeve (ukoliko postoje) koji su povezani s izvođenjem softvera i okruženjem u kojem se softver izvodi. To može uključivati zahtjeve vezane uz fizičko okruženje u kojem se nalazi sustav (glasno okruženje, jako osvjetljenje, prašina i sl.), drugi postojeći sustavi unutar kojih se softver treba izvoditi ili s kojima treba imati interakciju (operacijski sustav, drugi softver od kojeg preuzimamo podatke ili ih šaljemo).
5.4.5.	Sigurnost i privatnost	Navedite zahtjeve (ukoliko postoje) koji su povezani sa pitanjima sigurnosti i privatnosti podataka, te standardima i propisima vezanima uz tu problematiku. To može uključivati zahtjeve za korištenjem propisanih sigurnosnih procedura, tehnologija, usklađenost sa pravnim okvirima i sl.
5.5.	Skice zaslona	Vizualizirajte značajke interakcije između krajnjeg korisnika i softverskog rješenja kroz skice zaslona (engl. wireframe). Svrha skica je da na vizualan način predočimo i komuniciramo što aplikacija treba raditi, a ne da izradimo realan dizajn grafičkog sučelja. Pri tome skice možete na bilo koji način nacrtati (npr. ručno na papiru + slikanje s mobitelom, MS Paint, MS Word, Excel…)
6.	Dizajn softverskog sustava	Opisati dizajn softverskog sustava koji će ispunjavati specificirane korisničke zahtjeve. Pri izradi modela strukture i ponašanja budućeg sustava, potrebno je voditi se principima objektne-orijentacije, te koristiti UML jezik za modeliranje. Za izradu podatkovnih modela potrebno je koristiti ERA notaciju. Za razliku od skice arhitekture kreirane tijekom specifikacije korisničkih zahtjeva, ovdje je potrebno specificirati detaljnu i konačnu verziju arhitekture programskog rješenja sa svim podsustavima i komponentama te načinom njihove komunikacije i razmjene podataka. I u ovom poglavlju se može struktura preuzeti iz IEEE standarda ili drugih predložaka za dizajn sustava.
6.1	Funkcionalnost 1….	Za svaku funkcionalnost potrebno je odvojiti poglavlje koje prikazuje dizajn iste. Za navedenu funkcionalnost potrebno je navesti detaljnu specifikaciju slučaja korištenja, dijelove dijagrama podataka i dijagrama klasa koji se odnose na te funkcionalnosti, ali i sve ostale dijagrame i opise koji specificiraju arhitekturu, strukturu i ponašanje funkcionalnosti… Nerijetko su od pomoći i skice ekrana…
6….	Funkcionalnost …	
6…	Specifična tema 1…	Osim funkcionalnosti, pojedine druge teme su važne u fazi dizajna. To mogu biti teme kao što su: specifikacija API-ja, rad s bazom podataka, oporavak od pogreške ili druge teme koje su zajedničke za više funkcionalnosti… Ovakve teme je potrebno također posebno dizajnirati i u dokumentaciji opisati. Tim se onda obvezuje u svim

		funkcionalnostima koje se oslanjaju na ove teme implementaciju provesti na način kako je opisano u odgovarajućoj temi…
6…	Specifična tema …	
7.	Podaci o testiranju	Opisati na koji način je provedeno testiranje aplikacije. Koje metode testiranja su korištene? Koji dijelovi aplikacije su testirani? Koji scenariji su pokriveni? Ako aplikacija ima više uloga, te ako za uloge postoje podaci za logiranje potrebno je napraviti i eksplicitno prikazati login podatke za svaku pojedinu ulogu kojom se mogu testirati funkcionalnosti tih uloga.
8.	Podešavanje razvojne okoline	Ako projekt koristi specifične tehnologije i/ili biblioteke trećih strana, te ako se programski kod preuzet iz repozitorija ne može direktno kompajlirati i/ili pokrenuti, te ako je potrebno instalirati/podesiti bilo kakav dodatak, tehnologiju, alat, postavku, onda se podaci o podešavanju razvojne okoline specificiraju u ovom poglavlju.
9.	Uvođenje programa u rad	Opisati način instaliranja ili postavljanja programskog proizvoda kod korisnika. Ako je potrebno, specificirati postavljanje i podešavanje baze podataka ili upute za spajanje aplikacije na bazu i slično. U posebnom podpoglavlju navesti upute za deinstalaciju programa.
10.	Korisnička dokumentacija	Po dovršetku izrade aplikacije, dokumentirati (opisom i screenshot-ovima) osnovne scenarije korištenja aplikacije. Korisnička dokumentacija se smješta u sam programski proizvod i dostupna je tipkom F1, pri čemu se korisniku otvara ona dokumentacija koja se odnosi na kontekstu u kojem je kliknut F1. Ova dokumentacija može biti implementirana na više načina, a mi predlažemo CHM dokument koji se otvara pomoću koda.