Document de analiză a cerințelor clientului

Purificator Smart

Scopul acestui document este colectarea și comunicarea întregii echipe care sunt cerințele pe care le aveți de îndeplinit. Important de reținut că documentul nu se completează perfect secvențial. Scrieți ce credeți la un punct, iar după ce rezolvați și un al doilea punct, puteți reveni pentru a șlefui punctele anterioare. Este important să urmăriți o coerență globală a documentului, în urma acestor reveniri. Porniți de la acest template, dar nu păstrați aceste instrucțiuni în documentul final.

Scopul aplicației:

Identificați care este scopul principal al aplicației pe care o veți dezvolta. Pentru construirea scopului, veți începe de la cerințele de proiect din Curs I, dar îmbinat cu specificul aplicației pe care vă propuneți să îl faceți.

Scopul aplicației este de a îmbunătăți calitatea vieții prin reducerea expunerii la niveluri ridicate de umiditate, poluare și praf și crearea unui spațiu sigur prin detectarea unor niveluri periculoase de fum și de gaz. De asemenea, device-ul își propune și stimularea unei atmosfere ambientale care să favorizeze relaxarea și productivitatea.

Obiectivele aplicației:

Device-ul își propune să îndeplinească următoarele obiective:

1. detectarea nivelului de praf și înștiințarea utilizatorului
2. identificarea nivelului de umiditate și stabilizarea acestuia între 40-60%
3. detectarea nivelului de CO2 și afișarea de notificări în situațiile în care nivelul depășește 1000 PPM
4. alertarea în situații deosebite precum acumularea de fum sau scurgeri de gaz
5. emanarea de uleiuri esențiale pentru relaxare
6. crearea unei atmosfere plăcute prin jocuri de lumini
7. pornirea automată pe baza unui program stabilit

Definirea unui număr mic de obiective care acoperă cele mai importante perspective ale aplicației. Puteți începe de la cum ar arăta un proiect minim viabil (eventual pornind de la cerințele din curs). Ce valoare aduce clientului? Ce particularități de dezvoltare trebuie avute în vedere? Ce metrici de evaluare sunt relevante?

În general scopul și obiectivele se construiesc împreună; scopul fiind o privire de ansamblu a proiectului (și deci a obiectivelor) iar obiectivele încearcă să clarifice care sunt livrabilele, care împreună definesc scopul.

Grupul țintă

Utilizatori diferiți: persoane cu probleme de respirație, părinți, angajați remote, persoane cu alergii, utilizatorul de bază

1. Ca părinte, îmi doresc să pot seta un program purificatorului pentru a se potrivi cu programul de somn al copilului meu.
2. Ca persoană cu probleme de respirație, îmi doresc să pot controla nivelul de umiditate al aerului și de CO2 pentru a crea un spațiu sănătos.
3. Ca utilizator de bază, îmi doresc să fiu înștiințat de anumiți parametrii ai aerului pentru a preveni transmiterea virusurilor.
4. Ca persoană cu alergii, îmi doresc să monitorizez nivelul de praf pentru a preveni reacții alergice.
5. Ca utilizator de bază, îmi doresc să primesc notificări cu privire la nivelul crescut de CO2 pentru a putea crea ventilație în cameră.
6. Ca angajat remote, îmi doresc să pot crea un mediu ambiental plăcut prin utilizarea de uleiuri esențiale și prin jocuri de lumini pentru a mă destresa.
7. Ca utilizator de bază, îmi doresc să pot asculta muzică pentru a mă simți bine.
8. Ca utilizator de bază, îmi doresc să fiu anunțat cu privire la situații periculoase precum scurgere de gaz sau emanare de fum pentru a fi în siguranță.
9. Ca părinte, îmi doresc să pot pune muzică copilului pentru a putea adormi mai bine și a avea un somn liniștitor.

Cui îi adresați această aplicație? Care este profilul utilizatorului a cărui nevoie căutați să o satisfaceți? În special trebuie să surprindeți felurile diferite în care utilizatori diferiți folosesc același feature (de exemplu, cum folosește un gamer numpad-ul, și cum îl folosește un contabil) Folosiți User Stories.

Colectarea cerințelor

- autentificare în aplicație
- logare în aplicație
- oferirea unei experiențe personalizate de utilizare aplicației în funcție de nevoile utilizatorului
- crearea unui profil al utilizatorului (va răspunde la diverse întrebări: vârstă, sex, oraș, afecțiuni, program de funcționare al device-ului)
- detectarea nivelurilor de fum, CO2, umiditate și praf
- integrare de API pentru fum, nivel CO2, umiditate și praf
- colectare de date pentru persoane cu diferite probleme de sănătate
- integrarea unor campuri de completat date medicale în cadrul unei pagini de profil
- crearea de moduri speciale în funcție de anotimp, probleme de sănătate, momentul zilei
- folosirea unor funcții ce modifica setările dispozitivului în funcție de parametrii introduși

- afișarea unor notificări pentru indici (CO2, praf)
- introducerea unor metode care afișează diferiți indici în momentul în care anumiți parametrii depășesc nivelurile normale
- declanșarea unor semnale sonore în caz de pericol
- integrarea unor funcții care trimit mai departe anumite fișiere audio
- crearea unor jocuri de lumina
- trimiterea de date RGB către o componenta LED

Care sunt cerințele pe care userii din story-urile de mai sus le-ar cere? Care sunt cerințele de sistem care apar ca o consecință a cerințelor userului (performanța în anumiți parametrii; utilizarea anumitor resurse externe; utilizarea unui mod specific de dezvoltare ș.a.m.d.). Această zonă este bine completată dacă are un număr cât mai mare de cerințe (relevante), chiar dacă sunt prea multe, sau depășesc un pic dimensiunea proiectului pe care îl avem în vedere. Aici trebuie să înțelegem tot ce ***s-ar putea*** *face.*

Interpretarea și prioritizarea cerințelor

Dintre cerințele de mai sus vom interpreta și prioritiza cerințele.

1. Label-uiți cerințele funcționale / non-funcționale. Cerințele funcționale sunt cele care îndeplinesc o nevoie care a reieșit dintr-un user story, și răspund la întrebarea ce trebuie aplicația să facă. Cerințele nonfuncționale sunt cele care descriu calitățile de sistem, și răspund întrebărilor de tipul cum trebuie să fie un anumit feature sau aplicația cu totul? Acest labeling e suficient să îl faceți în documentul de analiză, nu e necesar să îl cărați pe mai departe.

Funcționale:

- oferirea unei experiențe personalizate de utilizare aplicației în funcție de nevoile utilizatorului
- detectarea nivelurilor de fum, CO2, umiditate și praf
- colectare de date pentru persoane cu diferite probleme de sănătate
- crearea de moduri speciale în funcție de anotimp, probleme de sănătate, momentul zilei
- afișarea unor notificări pentru indici (CO2, praf)
- declanșarea unor semnale sonore în caz de pericol
- crearea unor jocuri de lumina

Nefuncționale:

- înregistrare în aplicație
- logare în aplicație
- crearea unui profil al utilizatorului (va răspunde la diverse întrebări: vârstă, sex, oraș, afecțiuni, program de funcționare al device-ului)
- integrare de API pentru fum, nivel CO2, umiditate și praf
- integrarea unor campuri de completat date medicale în cadrul unei pagini de profil
- folosirea unor funcții ce modifica setările dispozitivului în funcție de parametrii introduși
- introducerea unor metode care afișează diferiți indici în momentul în care anumiți parametrii depășesc nivelurile normale

- integrarea unor funcții care trimit mai departe anumite fișiere audio
- trimiterea de date RGB către o componenta LED

detectarea nivelurilor de fum, CO2, umiditate și praf
- colectare de date pentru persoane cu diferite probleme de sănătate

2. Gruparea cerințelor

Identificați criterii de gruparea cerințelor care ulterior credeți că vă vor ajuta la dezvoltare.

- Folosiți criterii pentru a grupa cerințele într-un mod care vi se pare util – după zona de tehnologie (BE, DevOps ș.a.) după eventualele module ale app (comunicare, procesare, stocarea datelor ș.a.), după feature-uri. Menționați aceste categorii și în documentul de analiză.

Grupări:

- **stocarea datelor;**
- detectarea nivelurilor de fum, CO2, umiditate și praf
- colectare de date pentru persoane cu diferite probleme de sănătate
- crearea unui profil al utilizatorului (va răspunde la diverse întrebări: vârstă, sex, oraș, afecțiuni, program de funcționare al device-ului)
- integrarea unor campuri de completat date medicale în cadrul unei pagini de profil
- **procesare;**
- detectarea nivelurilor de fum, CO2, umiditate și praf
- colectare de date pentru persoane cu diferite probleme de sănătate
- **interpretare;**
- oferirea unei experiențe personalizate de utilizare aplicației în funcție de nevoile utilizatorului
- folosirea unor funcții ce modifica setările dispozitivului în funcție de parametrii introduși
- **comunicare;**
- declanșarea unor semnale sonore în caz de pericol
- crearea unor jocuri de lumina
- integrarea unor funcții care trimit mai departe anumite fișiere audio
- trimiterea de date RGB către o componenta LED
- **afișarea datelor;**
- afișarea unor notificări pentru indici (CO2, praf)
- introducerea unor metode care afișează diferiți indici în momentul în care anumiți parametrii depășesc nivelurile normale
- **dezvoltare;**

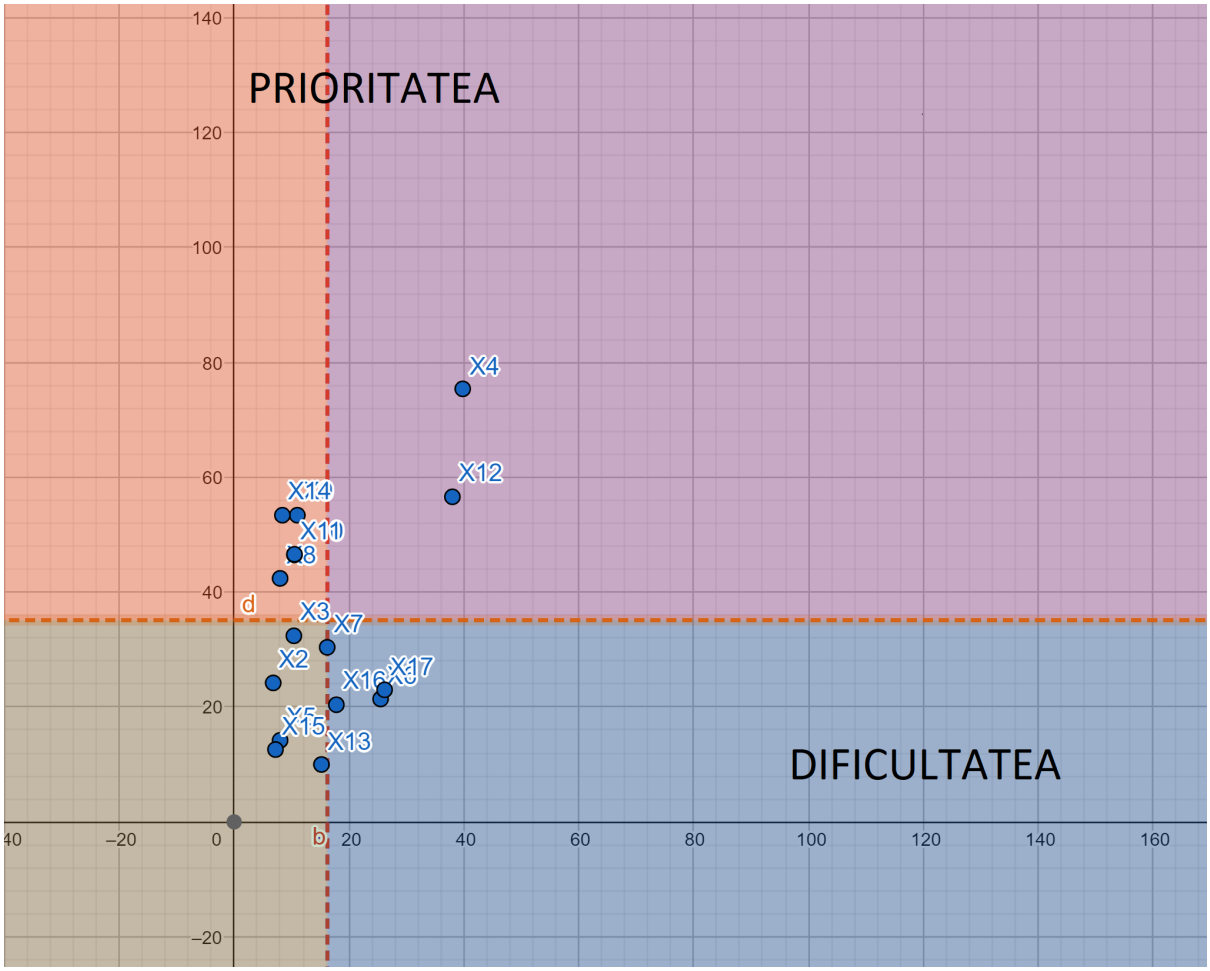
- înregistrare în aplicație
- logare în aplicație
- crearea unui profil al utilizatorului (va răspunde la diverse întrebări: vârstă, sex, oraș, afecțiuni, program de funcționare al device-ului)
- crearea de moduri speciale în funcție de anotimp, probleme de sănătate, momentul zilei
- integrare de API pentru fum, nivel CO2, umiditate și praf
- **testare;**

3. La acest moment puteți să creați un repo de GitHub pentru proiectul vostru și să puneți cerințele într-un back-log de issue-uri în GitHub Issues. Folosiți grupările de la 2 pentru a crea label-uri relevante pentru fiecare issue.

Resources:
The Product Backlog: A Step-by-step Guide În general sunt bune articolele de la toptal pe software engineering.

4. Play planning poker. Planning poker e un joc prin care toți membrii echipei evaluează prioritatea și dificultatea taskurilor în raport cu abilitățile individuale. Media acestor evaluări generează nivelul de prioritate și dificultate final al cerinței.
Recomand aplicația de aici: scrumpoker.online (pare că are și o integrare cu GitHub, dar nu am testat-o)
Pentru fiecare issue, veți juca câte două runde de planning poker. Trebuie să fiți într-un call. După ce alegeți issue-ul, fiecare user se autentifică în aplicație, și simultan ar trebui să votați dificultatea acelui issue. Faceți media și notați-o. Discutați dacă considerați că este cazul, și feel free să schimbați rezultatul dacă vi se pare că nu e potrivit.
Repetați pentru a identifica prioritatea taskului (adică cât de valoros este pentru aplicația finală). Notați și acest rezultat la fiecare task.

5. Plot the issues.
Realizați o axă, unde pe una dintre axe aveți dificultataea, iar pe cealaltă, prioritatea. Împărțiți axa în 4 cadrane (usor-valoros, dificil-valoros, usor-nevaloros, dificil-nevaloros). Astfel toate cerințele vor fi grupate în 4 categorii. Veți prioritiza cerințele usor-valoros, veți pune în backlog cele usor-nevaloroase și fidicil-valoroase, și în nice-to-have cele dificil nevaloros.



6. Check the features.
Verificați că adunate toate issue-urile prioritare, și din backlog, adunat constituie cele 5 feature-uri minim necesare pentru a îndeplini proiectul.

7. Add technical issues.
Adăugați issue-uri doar în GitHub de care va trebui să vă ocupați care nu reies din cerințe (dev setup, deployment setup, etc.)

Alocarea rolurilor

Fie alocați pentru fiecare cerință unul dintre membrii echipei care va realiza acea cerință, fie asociați membrii cu anumite label-uri, urmând ca respectivii membrii ai echipei să realizeze taskuri din labelul asociat.

Documentul de analiză va fi adăugat în GitHub-ul proiectului.