**Shembuj të rinj për FURPS**(disa I kam ne fletore)

**🔹 F – Functionality (Funksionale)**

1. Sistemi duhet të lejojë përdoruesin të ndryshojë fjalëkalimin.
2. Përdoruesi mund të shtojë produkte në shportë.
3. Sistemi duhet të ofrojë raport mujor të shitjeve për administratorin.
4. Përdoruesi mund të anulojë një porosi para se të dërgohet.
5. Sistemi duhet të lidhet me API-n e bankës për pagesa online.

**🔹 U – Usability (Përdorshmëria)**

1. Ndërfaqja duhet të ketë ikonat e qarta dhe të lexueshme.
2. Sistemi duhet të ketë tutorial të shkurtër për përdoruesit e rinj.
3. Aplikacioni duhet të mbështesë “drag and drop” për ngarkimin e dokumenteve.
4. Dizajni duhet të jetë responsive (përshtatet me telefon, tablet, desktop).
5. Sistemi duhet të ketë shortcut-e me tastierë për përdoruesit avancuar.

**🔹 R – Reliability (Besueshmëria)**

1. Sistemi nuk duhet të ketë më shumë se 0.5% dështim gjatë pagesave.
2. Në rast dështimi të serverit, sistemi duhet të kalojë automatikisht në server rezervë.
3. Backup-et duhet të ruhen për të paktën 30 ditë.
4. Sistemi duhet të gjenerojë alarm nëse databaza nuk përgjigjet për më shumë se 10 sekonda.
5. Sistemi nuk duhet të lejojë humbje të të dhënave të porosive.

**🔹 P – Performance (Performanca)**

1. Sistemi duhet të përgjigjet ndaj kërkesave brenda 0.8 sekondash.
2. Aplikacioni duhet të mbajë deri në 10,000 produkte në databazë pa ngadalësim.
3. Koha e gjenerimit të faturave nuk duhet të kalojë 2 sekonda.
4. Sistemi duhet të përballojë 200 transaksione në minutë.
5. Ngarkimi i aplikacionit mobil nuk duhet të kalojë 3 sekonda.

**🔹 S – Supportability (Mbështetje / Mirëmbajtje)**

1. Sistemi duhet të ketë API të dokumentuar për integrim me aplikacione të tjera.
2. Duhet të ketë testim automatik për çdo funksionalitet të ri.
3. Sistemi duhet të lejojë shtimin e moduleve të reja pa ndryshuar kodin ekzistues.
4. Çdo ndryshim në kod duhet të regjistrohet në një sistem versionimi (Git).

Manuali i përdoruesit dhe i administratorit duhet të jetë i përditësuar.  
  
  
**📊 Shembuj vetëm për MoSCoW  
Must (M)** → Kërkesa jetike, pa të sistemi nuk ekziston.

**Should (S)** → E rëndësishme, por mund të vonohet pak.

**Could (C)** → Bonus, opsionale.

**Won’t (W)** → Nuk do të bëhet tani, ndoshta më vonë.

1. **🔴 M – Must have (duhet të ketë, pa këto sistemi nuk punon)**
2. Përdoruesi duhet të bëjë login me email dhe fjalëkalim.
3. Sistemi duhet të ruajë të dhënat e blerjes në databazë.
4. Procesi i pagesës duhet të përfundojë pa gabime.
5. Sistemi duhet të ofrojë konfirmim pas çdo porosie.
6. Sistemi duhet të ketë opsion logout për siguri.
7. Sistemi duhet të mbështesë pagesat me kartelë krediti.
8. Sistemi duhet të jetë i disponueshëm online 24/7.
9. Sistemi duhet të ruajë backup minimalisht një herë në ditë.
10. Sistemi duhet të mbrojë të dhënat me enkriptim.
11. Sistemi duhet të funksionojë në shfletues modernë (Chrome, Firefox, Edge).
12. **🟠 S – Should have (e rëndësishme, por mund të vonohet pak)**
13. Përdoruesi duhet të mund të ndryshojë fjalëkalimin nga profili.
14. Sistemi duhet të ketë opsion për rikuperim fjalëkalimi me email.
15. Aplikacioni duhet të mbështesë dy gjuhë (shqip dhe anglisht).
16. Sistemi duhet të ketë raportim javor për administratorin.
17. Sistemi duhet të përballojë 1000 përdorues njëkohësisht.
18. Përdoruesi duhet të mund të filtrojë produktet sipas kategorive.
19. Sistemi duhet të mbështesë pagesa me PayPal.
20. Sistemi duhet të ruajë historikun e porosive të përdoruesit.
21. Aplikacioni duhet të ketë një “tutorial” për përdoruesit e rinj.
22. Sistemi duhet të ofrojë mbështetje teknike përmes chat-it live.
23. **🟡 C – Could have (opsionale, vetëm shtojnë vlerë)**
24. Aplikacioni mund të ketë dark mode.
25. Përdoruesi mund të personalizojë faqen kryesore me tema ngjyrash.
26. Sistemi mund të ketë rekomandime produktesh sipas interesave.
27. Përdoruesi mund të shtojë produkte në listën e dëshirave.
28. Aplikacioni mund të ketë opsion për të ndarë produktet në rrjete sociale.
29. Sistemi mund të ofrojë notifikime “push” në celular.
30. Përdoruesi mund të ruajë metodat e preferuara të pagesës.
31. Sistemi mund të ketë integrim me Google Maps për shpërndarjen.
32. Aplikacioni mund të ketë gamification (badge, pikë shpërblimi).
33. Përdoruesi mund të zgjedhë gjuhë të tretë (p.sh. gjermanisht).
34. **⚫ W – Won’t have (nuk do të përfshihen këtë herë)**
35. Sistemi nuk do të mbështesë integrim me Apple Watch në këtë version.
36. Aplikacioni nuk do të ketë kontroll me zë (voice commands) këtë vit.
37. Sistemi nuk do të mbështesë realitet të shtuar (AR) në këtë fazë.
38. Nuk do të ketë pagesa me kriptovaluta në versionin e parë.
39. Sistemi nuk do të ofrojë dërgesa ndërkombëtare në këtë fazë.
40. Aplikacioni nuk do të ketë version për televizorë smart.
41. Sistemi nuk do të ketë integrim me rrjete sociale në këtë fazë.
42. Nuk do të ketë mbështetje për pajisje shumë të vjetra (Windows XP).
43. Sistemi nuk do të ketë aplikacion desktop në këtë version.
44. Aplikacioni nuk do të ketë opsion offline në këtë fazë.

Ushtrime   
**📊 Tabela Testuese (FURPS + MoSCoW)**

| **#** | **Kërkesa** | **FURPS** | **Funksionale / Jo-funksionale** | **MoSCoW** |
| --- | --- | --- | --- | --- |
| 1 | Përdoruesi mund të krijojë llogari me email/fjalëkalim | F – Functionality | Funksionale | **M – Must** |
| 2 | Sistemi dërgon email konfirmimi pas blerjes | F – Functionality | Funksionale | **M – Must** |
| 3 | Admini mund të fshijë komente të papërshtatshme | F – Functionality | Funksionale | **M – Must** |
| 4 | Aplikacioni duhet të ketë dark mode | U – Usability | Jo-funksionale | **C – Could** |
| 5 | Aplikacioni duhet të jetë i qasshëm për përdorues me aftësi të kufizuara | U – Usability | Jo-funksionale | **S – Should** |
| 6 | Ndërfaqja duhet të jetë e thjeshtë që përdoruesi i ri të mësojë për 5 min | U – Usability | Jo-funksionale | **S – Should** |
| 7 | Sistemi duhet të rikuperohet brenda 30 sekondave pas dështimit | R – Reliability | Jo-funksionale | **S – Should** |
| 8 | Sistemi duhet të ketë backup automatik çdo 24 orë | R – Reliability | Jo-funksionale | **S – Should** |
| 9 | Disponueshmëri 99.9% mujore (uptime) | R – Reliability | Jo-funksionale | **M – Must** |
| 10 | Koha e ngarkimit të faqes < 2 sekonda | P – Performance | Jo-funksionale | **M – Must** |
| 11 | Sistemi përballon 1000 përdorues njëkohësisht | P – Performance | Jo-funksionale | **S – Should** |
| 12 | Gjenerimi i raporteve PDF < 3 sekonda | P – Performance | Jo-funksionale | **S – Should** |
| 13 | Aplikacioni përditësohet automatikisht në versionin më të ri | S – Supportability | Jo-funksionale | **S – Should** |
| 14 | Sistemi duhet të ketë log për gabime | S – Supportability | Jo-funksionale | **S – Should** |
| 15 | Kodi duhet të jetë modular për mirëmbajtje të lehtë | S – Supportability | Jo-funksionale | **S – Should** |
| 16 | Sistemi duhet të integrohet me PayPal | F – Functionality | Funksionale | **M – Must** |
| 17 | Përdoruesi mund të kërkojë produkte me fjalë kyçe | F – Functionality | Funksionale | **M – Must** |
| 18 | Sistemi duhet të mbështesë gjuhën shqipe dhe angleze | U – Usability | Jo-funksionale | **C – Could** |
| 19 | Ndihma online (help) duhet të jetë e disponueshme në çdo faqe | U – Usability | Jo-funksionale | **C – Could** |
| 20 | Sistemi nuk do të mbështesë integrim me Apple Watch në këtë version | — | — | **W – Won’** |

| **#** | **Kërkesa** | **FURPS** | **Funksionale / Jo-funksionale** | **MoSCoW** |
| --- | --- | --- | --- | --- |
| 1 | Ndryshim fjalëkalimi | F | Funksionale | M |
| 2 | Shto produkte në shportë | F | Funksionale | M |
| 3 | Raport mujor për admin | F | Funksionale | S |
| 4 | Anulim porosie | F | Funksionale | M |
| 5 | Integrim me API banke | F | Funksionale | M |
| 6 | Ikona të qarta | U | Jo-funksionale | C |
| 7 | Tutorial për përdorues | U | Jo-funksionale | C |
| 8 | Drag & drop dokumente | U | Jo-funksionale | C |
| 9 | Dizajn responsive | U | Jo-funksionale | M |
| 10 | Shortcut tastiere | U | Jo-funksionale | C |
| 11 | Dështim ≤ 0.5% pagesash | R | Jo-funksionale | M |
| 12 | Kalim automatik në server rezervë | R | Jo-funksionale | S |
| 13 | Backup për 30 ditë | R | Jo-funksionale | S |
| 14 | Alarm kur DB s’përgjigjet > 10s | R | Jo-funksionale | S |
| 15 | Pa humbje porosish | R | Jo-funksionale | M |
| 16 | Përgjigje < 0.8s | P | Jo-funksionale | M |
| 17 | 10,000 produkte pa ngadalësim | P | Jo-funksionale | S |
| 18 | Gjenerim fature < 2s | P | Jo-funksionale | S |
| 19 | 200 transaksione/min | P | Jo-funksionale | S |
| 20 | Aplikacion mobil < 3s | P | Jo-funksionale | M |
| 21 | API e dokumentuar | S | Jo-funksionale | S |
| 22 | Testim automatik | S | Jo-funksionale | S |
| 23 | Module të reja pa ndryshim kodi | S | Jo-funksionale | S |
| 24 | Versionim me Git | S | Jo-funksionale | M |
| 25 | Manual i përditësuar | S | Jo-funksionale | C |

Ligjerata 9  
**Pyetje të Mundshme si në Provim**

**1. Waterfall**

Cila është karakteristika kryesore e modelit Waterfall?  
A) Lejon ndryshime në çdo fazë të zhvillimit  
B) Kërkon dokumentim të plotë dhe rrjedhë lineare të fazave  
C) Dorëzon sistemin pjesë-pjesë  
D) Përdoret kryesisht për projekte të vogla pa dokumentim

**Përgjigjja: B**

**2. V-Model**

Pse përdoret modeli V?  
A) Për të kursyer kohën duke mos testuar fare  
B) Për të testuar vetëm në fund të zhvillimit  
C) Për të lidhur çdo fazë të zhvillimit me një fazë testimi  
D) Sepse nuk kërkon dokumentim fare

**Përgjigjja: C**

**3. Incremental**

Cila është veçoria kryesore e modelit inkremental?  
A) Dorëzimi i funksionaliteteve pjesë-pjesë dhe testimi i tyre i menjëhershëm  
B) Zhvillimi i plotë i sistemit dhe dorëzimi vetëm në fund  
C) Testimi kryhet vetëm një herë, në fazën finale  
D) Nuk pranon ndryshime gjatë procesit

**Përgjigjja: A**

**4. Iterative**

Si dallon modeli iterativ nga modeli inkremental?  
A) Iterativi dorëzon funksione të reja, inkrementali vetëm përmirëson të vjetrat  
B) Iterativi përmirëson versionin ekzistues vazhdimisht, inkrementali shton module të reja  
C) Iterativi s’ka testim, inkrementali ka  
D) Iterativi përdoret vetëm në projekte të vogla

**Përgjigjja: B**

**5. Spiral**

Cila është veçoria unike e modelit Spiral?  
A) Shpejtësia e zhvillimit  
B) Analiza e rrezikut në çdo cikël  
C) Përdorimi i komponentëve të gatshëm  
D) Dokumentimi i thjeshtë

**Përgjigjja: B**

**6. Prototyping**

Kur përdoret modeli Prototyping?  
A) Kur kërkesat janë të qarta që në fillim  
B) Kur duam të kuptojmë më mirë kërkesat e paqarta  
C) Kur s’kemi nevojë për komunikim me përdoruesin  
D) Kur duam të shmangim testimin

**Përgjigjja: B**

**7. RAD**

Cili është përfitimi kryesor i RAD?  
A) Koha e shkurtër e zhvillimit me përdorim të prototipeve dhe komponentëve ekzistues  
B) Dokumentim i plotë dhe rigoroz  
C) Testim vetëm në fund të projektit  
D) Mungesa e komunikimit me klientin

**Përgjigjja: A**

**8. Agile**

Cila është filozofia bazë e Agile?  
A) Të bëjmë dokumentim të plotë përpara kodimit  
B) Të zhvillojmë pa komunikim me klientin  
C) Dorëzime të shpejta dhe përshtatje ndaj ndryshimeve  
D) Testim vetëm në fund

**Përgjigjja: C**

**Pyetje të tipit “krahasim” (stili i profesorit)**

**1. Waterfall vs Incremental**

Cila nga deklaratat dallon më së miri modelin Inkremental nga Waterfall?  
A) Të dy modelet nuk përdorin fare testimin gjatë zhvillimit  
B) Inkremental dorëzon funksionalitete pjesë-pjesë, ndërsa Waterfall kërkon dorëzimin e plotë në fund  
C) Waterfall është më fleksibël ndaj ndryshimeve se Inkremental  
D) Inkremental nuk kërkon dokumentim, ndërsa Waterfall po

**Përgjigjja: B**

**2. Waterfall vs Iterative**

Si ndryshon Iterativi nga Waterfall?  
A) Iterativi e zhvillon sistemin në përsëritje, ndërsa Waterfall e zhvillon njëherë për gjithmonë  
B) Iterativi dorëzon vetëm në fund, ndërsa Waterfall pjesë-pjesë  
C) Iterativi nuk ka nevojë për feedback, ndërsa Waterfall po  
D) Waterfall është gjithmonë më fleksibël se Iterativi

**Përgjigjja: A**

**3. Incremental vs Iterative**

Cili është dallimi kryesor midis Inkremental dhe Iterativ?  
A) Iterativi shton module të reja, ndërsa Inkrementali përmirëson ato ekzistueset  
B) Inkrementali shton module të reja funksionale, ndërsa Iterativi përmirëson versionin ekzistues  
C) Të dy janë të njëjtë, vetëm me emër të ndryshëm  
D) Iterativi s’ka testim, ndërsa Inkrementali ka

**Përgjigjja: B**

**4. Spiral vs Waterfall**

Pse modeli Spiral konsiderohet më i përshtatshëm për projekte të mëdha se Waterfall?  
A) Sepse Spiral ka kosto më të ulët  
B) Sepse Spiral përfshin analiza të rrezikut në çdo cikël, ndërsa Waterfall jo  
C) Sepse Waterfall dorëzon pjesë-pjesë funksionalitetet, ndërsa Spiral vetëm në fund  
D) Sepse Spiral nuk kërkon planifikim

**Përgjigjja: B**

**5. Spiral vs Iterative**

Çfarë e dallon Spiral nga Iterative?  
A) Spiral përfshin menaxhimin e rrezikut në çdo iteracion, ndërsa Iterative jo  
B) Spiral nuk lejon ndryshime, Iterative po  
C) Spiral dorëzon vetëm në fund, Iterative pjesë-pjesë  
D) Spiral është më pak kompleks se Iterative

**Përgjigjja: A**

**6. Prototyping vs Waterfall**

Kur është më i përshtatshëm modeli Prototyping sesa Waterfall?  
A) Kur kërkesat janë të qarta dhe të pandryshueshme  
B) Kur kërkesat janë të paqarta dhe duhen sqaruar me përdoruesin  
C) Kur duam dorëzime lineare pa ndryshime  
D) Kur projekti është shumë i madh dhe kritik

**Përgjigjja: B**

**7. Prototyping vs Incremental**

Cili është dallimi kryesor midis Prototyping dhe Incremental?  
A) Prototyping përdoret për të kuptuar kërkesat, Incremental për të ndërtuar sistemin gradualisht  
B) Prototyping dorëzon module funksionale një nga një, Incremental vetëm një prototip jo-funksional  
C) Prototyping nuk kërkon komunikim me përdoruesin, Incremental po  
D) Incremental është më i mirë kur kërkesat janë të paqarta

**Përgjigjja: A**

**8. RAD vs Waterfall**

Pse RAD është më i përshtatshëm se Waterfall për projekte me afate të shkurtra?  
A) Sepse RAD përdor prototipe dhe komponentë ekzistues, duke e përshpejtuar zhvillimin  
B) Sepse RAD nuk kërkon testim fare  
C) Sepse Waterfall nuk përdor dokumentim  
D) Sepse RAD nuk përfshin fare përdoruesin

**Përgjigjja: A**

**9. Agile vs Waterfall**

Cili është ndryshimi kryesor mes Agile dhe Waterfall?  
A) Agile është fleksibël dhe përshtatet ndaj ndryshimeve, ndërsa Waterfall është i ngurtë  
B) Agile nuk kërkon testim, ndërsa Waterfall po  
C) Agile dokumenton më shumë se Waterfall  
D) Waterfall është më i mirë për kërkesa që ndryshojnë vazhdimisht

**Përgjigjja: A**

**10. Agile vs RAD**

Cili është dallimi kryesor midis Agile dhe RAD?  
A) Agile bazohet në iteracione të shkurtra dhe komunikim të vazhdueshëm, ndërsa RAD në përdorim të prototipeve dhe komponentëve ekzistues  
B) Agile dorëzon vetëm një prototip, RAD dorëzon inkremente të shumta  
C) RAD është më fleksibël ndaj ndryshimeve sesa Agile  
D) Agile është vetëm për projekte të vogla, RAD për projekte të mëdha

**Përgjigjja: A**

**Waterfall**

1. Cila është veçoria kryesore e modelit Waterfall?  
   A) Dorëzon funksionalitete pjesë-pjesë  
   B) Ka rrjedhë lineare dhe dokumentim të plotë  
   C) Përdoret kur kërkesat janë të paqarta  
   D) Nuk kërkon testim fare  
   **Përgjigjja: B**
2. Kur është më i përshtatshëm modeli Waterfall?  
   A) Kur kërkesat janë të qarta dhe të pandryshueshme  
   B) Kur kërkesat ndryshojnë shpesh  
   C) Kur duam dorëzime të shpejta  
   D) Kur projekti është shumë kritik  
   **Përgjigjja: A**

**V-Model**

1. Çfarë e dallon V-Model nga Waterfall?  
   A) Nuk ka testim fare  
   B) Çdo fazë ka një fazë testimi përballë  
   C) Dorëzon funksionalitete pjesë-pjesë  
   D) Nuk kërkon dokumentim  
   **Përgjigjja: B**
2. Për çfarë përdoret më shumë V-Model?  
   A) Projekte të vogla pa rrezik  
   B) Sisteme kritike si aviacioni dhe mjekësia  
   C) Projekte me kërkesa të paqarta  
   D) Projekte pa dokumentim  
   **Përgjigjja: B**

**Incremental**

1. Cila është veçoria kryesore e modelit Inkremental?  
   A) Dorëzimi i funksionaliteteve pjesë-pjesë  
   B) Testim vetëm në fund  
   C) Nuk pranon ndryshime  
   D) Nuk kërkon dokumentim  
   **Përgjigjja: A**
2. Pse modeli Inkremental është më fleksibël se Waterfall?  
   A) Sepse pranon ndryshime gjatë zhvillimit  
   B) Sepse nuk ka testim  
   C) Sepse nuk kërkon planifikim fare  
   D) Sepse dorëzon vetëm në fund  
   **Përgjigjja: A**

**Iterative**

1. Cila është veçoria kryesore e modelit Iterativ?  
   A) Përmirëson versionin ekzistues me përsëritje  
   B) Dorëzon module të reja funksionale një nga një  
   C) Nuk ka testim fare  
   D) Është i njëjtë me Waterfall  
   **Përgjigjja: A**
2. Pse Iterativi është i përshtatshëm për projekte me kërkesa që evoluojnë?  
   A) Sepse nuk kërkon planifikim  
   B) Sepse lejon testim dhe ndryshim pas çdo iteracioni  
   C) Sepse dorëzon vetëm në fund  
   D) Sepse nuk ka dokumentim  
   **Përgjigjja: B**

**Incremental vs Iterative**

1. Çfarë e dallon Incremental nga Iterative?  
   A) Incremental shton module të reja, Iterative përmirëson versionin ekzistues  
   B) Iterative shton module, Incremental përmirëson  
   C) Të dy janë identikë  
   D) Njëri ka testim, tjetri jo  
   **Përgjigjja: A**

**Spiral**

1. Cila është veçoria kryesore e modelit Spiral?  
   A) Fokus në shpejtësi  
   B) Menaxhim i rrezikut në çdo iteracion  
   C) Nuk ka testim fare  
   D) Është më i thjeshtë se Waterfall  
   **Përgjigjja: B**
2. Për çfarë është më i përshtatshëm modeli Spiral?  
   A) Projekte të vogla me kërkesa të qarta  
   B) Projekte të mëdha dhe kritike me rrezik të lartë  
   C) Projekte pa dokumentim  
   D) Projekte pa testim  
   **Përgjigjja: B**

**Prototyping**

1. Kur përdoret modeli Prototyping?  
   A) Kur kërkesat janë të qarta që në fillim  
   B) Kur kërkesat janë të paqarta dhe duhen kuptuar  
   C) Kur duam dorëzime vetëm në fund  
   D) Kur nuk kemi komunikim me përdoruesin  
   **Përgjigjja: B**
2. Cili është rreziku kryesor i modelit Prototyping?  
   A) Përdoruesit mund ta ngatërrojnë prototipin me produktin final  
   B) Nuk ka testim  
   C) Nuk ka komunikim fare  
   D) Nuk mund të përdoret për kërkesa të paqarta  
   **Përgjigjja: A**

**RAD (Rapid Application Development)**

1. Cila është veçoria kryesore e RAD?  
   A) Zhvillim i shpejtë duke përdorur prototipe dhe komponentë ekzistues  
   B) Dokumentim i plotë dhe i detajuar  
   C) Dorëzim vetëm në fund  
   D) Mungesë e komunikimit me përdoruesin  
   **Përgjigjja: A**
2. Kur është më i përshtatshëm RAD?  
   A) Projekte të vogla/mesme me afate të shkurtra  
   B) Projekte kritike dhe shumë të mëdha  
   C) Projekte pa testim  
   D) Projekte me kërkesa të pandryshueshme  
   **Përgjigjja: A**

**Agile**

1. Cila është filozofia bazë e Agile?  
   A) Dorëzim i shpejtë dhe përshtatje ndaj ndryshimeve  
   B) Dokumentim i plotë dhe i ngurtë  
   C) Testim vetëm në fund  
   D) Mungesë komunikimi me klientin  
   **Përgjigjja: A**
2. Cilat janë metodologjitë më të njohura Agile?  
   A) Waterfall dhe V-Model  
   B) SCRUM, Kanban, XP  
   C) Spiral dhe Incremental  
   D) RAD dhe Prototyping  
   **Përgjigjja: B**

**Pyetje krahasuese (më të vështira)**

1. Waterfall vs Agile → Agile është më i mirë kur:  
   A) Kërkesat ndryshojnë shpesh  
   B) Kërkesat janë të qarta që në fillim  
   C) Projekti është shumë kritik  
   D) Nuk duam komunikim me klientin  
   **Përgjigjja: A**
2. V-Model vs Spiral → Spiral ndryshon sepse:  
   A) Thekson menaxhimin e rrezikut, V-Model jo  
   B) Nuk ka testim, ndërsa V-Model ka  
   C) Dorëzon vetëm në fund, ndërsa V-Model pjesë-pjesë  
   D) Është më i thjeshtë se V-Model  
   **Përgjigjja: A**
3. Incremental vs Prototyping → Prototyping përdoret:  
   A) Për të kuptuar kërkesat, ndërsa Incremental për ndërtim gradual  
   B) Për të dorëzuar module, ndërsa Incremental për të testuar versionet  
   C) Kur kërkesat janë të qarta, ndërsa Incremental për kërkesa të paqarta  
   D) Për projekte kritike, ndërsa Incremental për të vogla  
   **Përgjigjja: A**
4. Agile vs RAD → dallimi kryesor është:  
   A) Agile bazohet në iteracione dhe bashkëpunim, RAD në komponentë ekzistues  
   B) Agile nuk ka komunikim, RAD ka  
   C) RAD është më fleksibël se Agile  
   D) Të dy janë identikë  
   **Përgjigjja: A**
5. Iterative vs Spiral → Spiral shton:  
   A) Menaxhimin e rrezikut në çdo cikël  
   B) Dorëzim vetëm në fund  
   C) Dokumentim të tepërt  
   D) Mungesë testimi  
   **Përgjigjja: A**
6. Agile vs Waterfall → dallimi kryesor është:  
   A) Agile përshtatet ndaj ndryshimeve, Waterfall jo  
   B) Agile dorëzon vetëm në fund, Waterfall pjesë-pjesë  
   C) Agile nuk teston fare, Waterfall teston  
   D) Waterfall është më fleksibël se Agile  
   **Përgjigjja: A**
7. Prototyping vs Agile → Prototyping fokusohet në:  
   A) Kuptimin e kërkesave, ndërsa Agile në dorëzime të shpejta dhe të vazhdueshme  
   B) Dokumentim të plotë, ndërsa Agile pa dokumentim  
   C) Projekte kritike, ndërsa Agile për projekte të vogla  
   D) Dorëzime vetëm në fund  
   **Përgjigjja: A**
8. Spiral vs Incremental → Spiral është më i përshtatshëm sepse:  
   A) Menaxhon rrezikun në çdo iteracion  
   B) Nuk kërkon dokumentim  
   C) Nuk dorëzon funksionalitete pjesë-pjesë  
   D) Është më i shpejtë gjithmonë  
   **Përgjigjja: A**

**Avantazhe / Disavantazhe**

1. Avantazhi kryesor i Waterfall është:  
   A) Thjeshtësia dhe dokumentimi i plotë  
   B) Fleksibiliteti maksimal  
   C) Dorëzimi pjesë-pjesë  
   D) Mungesa e dokumentimit  
   **Përgjigjja: A**
2. Disavantazhi kryesor i Iterativ është:  
   A) Mund të marrë më shumë kohë sepse ka shumë përsëritje  
   B) Nuk lejon ndryshime  
   C) Nuk teston fare  
   D) Nuk përdoret për kërkesa që ndryshojnë  
   **Përgjigjja: A**
3. Avantazhi kryesor i Inkremental është:  
   A) Feedback i hershëm nga përdoruesi  
   B) Dorëzim vetëm në fund  
   C) Nuk kërkon dokumentim  
   D) Nuk pranon ndryshime  
   **Përgjigjja: A**
4. Disavantazhi kryesor i Spiral është:  
   A) Kompleksiteti dhe kosto e lartë  
   B) Nuk teston fare  
   C) Nuk menaxhon rrezikun  
   D) Është shumë i ngurtë  
   **Përgjigjja: A**
5. Avantazhi i Agile është:  
   A) Fleksibiliteti dhe përshtatja ndaj ndryshimeve  
   B) Dorëzim vetëm në fund  
   C) Nuk ka nevojë për komunikim  
   D) Dokumentim i tepërt  
   **Përgjigjja: A**

👉 Këto janë **30 pyetje kryesore**, dhe po i lë edhe **5 shtesë** që mund t’i përdorë profesori për “kurth”.

1. Cili model është më i mirë kur kërkesat janë të qarta dhe të pandryshueshme?  
   **Përgjigjja: Waterfall**
2. Cili model është më i mirë për projekte kritike si aviacioni dhe mjekësia?  
   **Përgjigjja: V-Model**
3. Cili model është më i mirë kur kërkesat janë të paqarta dhe ndryshojnë?  
   **Përgjigjja: Prototyping ose Agile**
4. Cili model është më i mirë për projekte të mëdha me rrezik të lartë?  
   **Përgjigjja: Spiral**
5. Cili model është më i mirë kur duam dorëzime të shpejta dhe fleksibilitet?  
   **Përgjigjja: Agile**

Detyra e profesorit-ligj 8 SQARIME   
**Si dallojmë Entity, Value Object dhe si i kthejmë kërkesat në klasë?**

**📌 1. Rregulli për Entity:**

* Ka **ID unike** (mund të ruhet në DB me primary key).
* Mund të ndryshojë gjatë kohës (ka jetë).
* Ka kuptim edhe më vete.  
  👉 Shembuj: User, Course, Enrollment, Product, Order.

**📌 2. Rregulli për Value Object:**

* **Nuk ka ID** (dy adresa të njëjta konsiderohen të barabarta).
* Nuk jeton pa entitetin.
* Është pjesë e ngushtë (composition).  
  👉 Shembuj: Address, PhoneNumber, MaterialFile, Price.

**📌 3. Rregulli për Aggregate Root:**

* Entiteti kryesor që menaxhon integritetin e një grupi objektesh.
* P.sh. Course është root që kontrollon Material dhe CourseStatus.

**🔹 Si ta lexosh kërkesën dhe ta kthesh në diagram?**

**Shembulli që the:**

👉 “Sistemi duhet të mundësojë **filtrimin e kurseve aktive dhe të mbyllura**.”

1. **Identifiko objektin bazë** → Kursi.
2. **Kërko atributin që kërkohet** → statusi i kursit.
3. **Vendos a është Entity apo Value Object?**
   * Statusi nuk jeton pa kurs, por mund të jetë **listë e paracaktuar** (Active, Closed).
   * Në këtë rast, profesori e ka modeluar si **Entity: CourseStatus**.
4. **Në class diagram**:
   * Kemi klasën Course me një atribut CourseStatus.
   * Course ── 1..1 ── CourseStatus.

**🔹 Procesi për çdo kërkesë funksionale**

1. **Lexo kërkesën**.
2. Pyet:
   * Kush është **entiteti kryesor**?
   * A është ky objekt me **ID** apo vetëm një tipar që lidhet me entitetin?
   * A ka kuptim më vete apo jo?
3. Vendos nëse është:
   * **Entity** (me ID)
   * **Value Object** (pa ID, pjesë e një entiteti)
   * **Relacion/Referencë** (nëse lidhet me një entitet tjetër në një modul tjetër).
4. Pastaj futi në **modulin përkatës (Bounded Context)**.
5. Kur disa kërkesa kanë të njëjtin entitet kryesor → bëhen një **Aggregate**.
6. Nxirr varësitë mes moduleve për **Package Diagram**.
7. Për **Class Diagram**, trego lidhjet (association, aggregation, composition, inheritance).

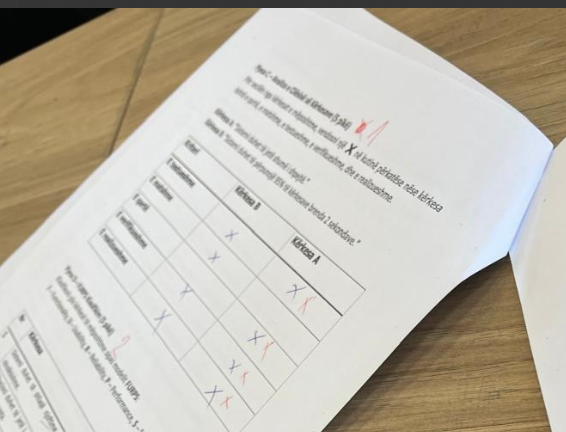
**🔹 Mini-shembull i zbërthimit të një kërkese**

👉 “Sistemi duhet të njoftojë studentët kur regjistrohen me sukses në një kurs.”

* **Entitet kryesor:** Enrollment.
* **Objekte të përfshira:** Student (referencë nga User Management), Course (referencë nga Course Management).
* **Funksionalitet:** event/operacion në Enrollment: NotifyStudent().
* **Në Class Diagram:**
  + Enrollment ─── Student (association)
  + Enrollment ─── Course (association)
  + Metodë notifyStudent() brenda Enrollment.

**🔹 Përmbledhje**

* **Entity** = gjithmonë me ID, ka jetë më vete.
* **Value Object** = pjesë e entitetit, pa ID, vdes me të.
* **Aggregate Root** = entiteti që kontrollon cluster-in.
* **Referenca** = association mes entiteteve të moduleve të ndryshme.
* **Nga kërkesat**: gjej entitetin kryesor + atributet + veprimet → futi në klasë.
* **Në package diagram**: trego kush bën **use** të kujt.

detyra e provimit ,e ki nje table edhe ki me vendos X nese kerkesa I perket asaj kolone   
  
**Analiza e dy kërkesave:**

**Kërkesa A: *"Sistemi duhet të jetë shumë i shpejt."  
(ne provim nuk e vendos*** ❌ ***por nuk venods asgje ,e nese eshte per*** ✅ e vendos ❌)

* ❌ Nuk është **e matshme** → “shumë i shpejt” nuk ka kriter numerik.
* ❌ Nuk është **e testueshme** → nuk mund ta testosh pa shifër.
* ❌ Nuk është **e qartë** → secili e interpreton ndryshe.
* ❌ Nuk është **e verifikueshme** → nuk ka standard për krahasim.
* ❌ **E realizueshme** në parim, por kërkesa është e dobët sepse nuk ka parametra.
* 📌 Kjo është një **kërkesë jo-funksionale**, por **e paformuluar mirë**.

**Kërkesa B: *"Sistemi duhet të përpunojë 95% të kërkesave brenda 2 sekondave."***

* ✅ **E matshme** → 95% + 2 sekonda janë parametra konkretë.
* ✅ **E testueshme** → mund të bësh test performancash.
* ✅ **E qartë** → kuptohet saktë çka kërkohet.
* ✅ **E verifikueshme** → mund të provohet me teste.
* ✅ **E realizueshme** (në varësi të teknologjisë).
* 📌 Kjo është një **kërkesë jo-funksionale e shkruar mirë**, sepse lidhet me performancën (si e bën sistemi).

detyra te ngjajshme me te provimit   
**Shembuj të Kërkesave**

**🔹 Kërkesë Funksionale (formuluar mirë)**

**“Sistemi duhet t’i lejojë përdoruesit të ndryshojë fjalëkalimin duke dërguar një kod verifikimi në email.”**

* E qartë ✅
* E matshme ✅ (mund të testosh nëse kodi dërgohet / pranohet)
* E testueshme ✅
* E verifikueshme ✅
* E realizueshme ✅

**🔹 Kërkesë Jo-funksionale (formuluar keq)**

**“Sistemi duhet të jetë i lehtë për përdoruesit.”**

* E qartë ❌ (çka nënkupton “i lehtë”?)
* E matshme ❌
* E testueshme ❌
* E verifikueshme ❌
* E realizueshme x

➡️ Kjo në fakt është **jo-funksionale e shkruar keq**, jo funksionale klasike.

**🔹 Kërkesë Jo-funksionale (formuluar mirë)**

**“Sistemi duhet të përballojë 1000 përdorues njëkohësisht pa rënë shpejtësia nën 3 sekonda për faqe.”**

* E qartë ✅
* E matshme ✅ (1000 përdorues, 3 sekonda)
* E testueshme ✅ (mund të bësh load test)
* E verifikueshme ✅
* E realizueshme ✅ (nëse infrastrukturë e mirë)

**🔹 Kërkesë Jo-funksionale (formuluar keq)**

**“Sistemi duhet të ketë performancë të lartë.”**

* E qartë ❌
* E matshme ❌
* E testueshme ❌
* E verifikueshme ❌
* E realizueshme x  
    
  **📌 Ushtrimet**
* **Kërkesa 1**
* “Sistemi duhet të lejojë përdoruesin të ruajë porosinë në databazë.”
* **Kërkesa 2**
* “Sistemi duhet të jetë shumë i sigurt.”
* **Kërkesa 3**
* “Sistemi duhet të gjenerojë një raport mujor për çdo klient, brenda 10 sekondave pas kërkesës.”
* **Kërkesa 4**
* “95% e transaksioneve duhet të përfundojnë brenda 5 sekondave, edhe kur janë aktivë mbi 500 përdorues.”
* **Kërkesa 5**
* “Sistemi duhet të ketë dizajn të bukur dhe tërheqës për përdoruesin.”
* **Kërkesa 6**
* “Sistemi duhet të dërgojë një email konfirmimi pas çdo regjistrimi të suksesshëm.”