



Optimalt lärande Arkitekturdokument

PUM09

2024-02-18

Version 1.0

Status

Granskad		
Godkänd		



Projektidentitet

Grupp E-post: emiho191@student.liu.se

Hemsida: <http://www.liu.se/grouppage>

Beställare: Kristian Sandahl, Linköpings universitet
Tfn: 013-28 19 57
E-post: kristian.sandahl@liu.se

Kund: Jörgen Blomvall, Linköpings universitet
Tfn: 013-28 14 06
E-post: jorgen.blomvall@liu.se

Handledare: Lena Buffoni
Tfn: 013-28 40 46
E-post: lena.buffoni@liu.se

Kursansvarig: Kristian Sandahl, Linköpings universitet
Tfn: 013-28 19 57
E-post: kristian.sandahl@liu.se

Projektdeltagare

Namn	Roll	E-post
Mabest Amin (MA)	Arkitekt	mabam091@student.liu.se
Odin Dahlström (OD)	Utvecklingsledare	odida723@student.liu.se
Emil Holmstedt (EH)	Teamledare	emiho191@student.liu.se
Martin Hultgren (MH)	Kvalitetssamordnare, Databasansvarig	marhu242@student.liu.se
Casper Erik Nerf Kanefall (CK)	Testledare	casne582@student.liu.se
Eric Van Nunen (EVN)	Analysansvarig	eriva185@student.liu.se
Wiliam Puranen (WP)	Konfigurationsansvarig	wilpu732@student.liu.se
Yadgar Suleiman (YS)	Dokumentansvarig	yadsu309@student.liu.se



INNEHÅLL

1	Inledning	1
1.1	Syfte	1
1.2	Översikt	1
2	Arkitekturmål och filosofi	2
3	Antaganden och beroenden	2
4	Arkitektoniska signifikanta krav	3
4.1	Krav från tidigare PUM-projekt	3
4.2	Nya krav från kunden	4
5	Val, begränsningar och motiveringar	5
6	Arkitektoniska mekanismer	6
7	Nyckelabstraktioner	6
8	Arkitektoniska designmönster	7
9	Arkitektoniska vyer	8
10	Referenser	12



DOKUMENTHISTORIK

Version	Datum	Utförda förändringar	Utförda av	Granskad
1.0	2024-02-19	Första utkastet.	MA, OD	EVN, EH



1 INLEDNING

1.1 Syfte

Syftet med arkitekturdokumentet är att möjliggöra en översiktlig uppföljning av utvecklingsprocessen från idé till implementering av den redan framställda applikationen. Dokumentet summerar beslut, krav och teknisk beskrivning genom hela processen för att genomgå projektet Optimalt lärande. Detta arkitekturdokument är avsett för projektgruppen samt för externa intressenter för att de tydligt ska kunna följa utvecklingsprocessen och i framtiden kunna bygga på applikationen genom att bygga på en bra förståelse för arkitekturen. Syftet med dokumentet är att beskriva de val och motiveringar som har gjorts och ska göras i utvecklingsprocessen. Det är även viktigt att identifiera vilka vägval som har prioriterats och vilka som har avförts från projektet och det huvudsakliga målet är att alla gruppmedlemmar ska förstå det önskade resultatet.

1.2 Översikt

Arkitekturmål och filosofi definierar de främsta målen för produkten/applikationen och dess tekniska grundprinciper. Antaganden och beroenden går djupare in på de förutsättningar och externa faktorer som påverkar produkten.

Arkitektoniska signifikanta krav beskriver de kraven som projektet har fått av kunden för produkten. Val, begränsningar och motiveringar förklarar de designval som projektgruppen har gjort, i detta inkluderas motiveringar för varför vissa funktioner och dessutom vilka begränsningar som har beaktats samt fastställts.

Arkitektoniska mekanismer beskriver främst de metoder och strategier som bör användas för att skapa en väloptimerad arkitektur över hela projektet även på det som redan gjorts av tidigare PUM-grupper. Nyckelabstraktioner ger en mer grundläggande beskrivning av de funktioner som ska implementeras.

Arkitektoniska designmönster beskriver de strukturer och lösningar som projektgruppen har valt för att hantera specifika utmaningar med arkitekturen. Vyer som representerar systemet visuellt, inklusive användning av ramverk och systemskisser, för att tydliggöra implementeringen samt logiken bakom dessa implementeringar.



2 ARKITEKTURMÅL OCH FILOSOFI

Arkitekturen för produkten bygger på ett tydligt gränssnitt mellan modulerna. Genom att adressera dessa krav kan arkitekturen möta behoven hos både kunden samt användaren genom att erbjuda valmöjligheter samtidigt som säkerheten upprätthålls.

Vidare är en viktig del av arkitekturen att säkerställa kommunikationen mellan mobilapplikationen och backend-systemet så att det sker effektivt, främst för att föra statistik och optimera inloggningsfunktionen. Genom att ha tydliga gränssnitt och klart definierade arkitekturmål samt krav kan arkitekturen säkerställa en smidig integration mellan de olika delsystemen.

De arkitekturmål och krav som har definierats inkluderar behovet av att hantera olika licenskrav, beroenden, säkerställa effektiv kommunikation mellan mobilapplikationen och backend-systemet samt att erbjuda en tydlig och säker systemarkitektur som kan hantera både nuvarande och framtida krav vilket är ett mål från kunden. Dessa mål formar arkitekturen och filosofin genom att fokusera på att möta specifika krav och behov som har identifierats i projektet från både kunden och användaren.

3 ANTAGANDEN OCH BEROENDEN

Projektet bygger på att återanvända arkitekturen från tidigare kandidatarbeten, vilket är en viktig faktor för de arkitektoniska besluten. Att bygga om hela systemet skulle vara både tidskrävande och ineffektivt, så återanvändning av befintlig arkitektur är fördelaktigt för projektet.

Gruppmedlemmarna behöver inte ha några specifika kunskaper för att delta i projektet, eftersom utbildning inom relevanta områden är planerad, såsom kunskaper inom ramverket *Flutter*. Detta påverkar hur arbetsuppgifterna fördelas och vilka områden som prioriteras för inläring inom gruppen.

Databasutveckling hanteras främst av en tillägnad databasansvarig och en vice-databasansvarig. Detta beroende påverkar hur resurserna allokeras och även hur arbetet delas upp inom gruppen för att effektivt hantera de olika kompetenserna och ansvarsområdena.



4 ARKITEKTONISKA SIGNIFIKANTA KRAV

4.1 Krav från tidigare PUM-projekt

Tabell 1: Tabell över krav från tidigare PUM-projekt.

Krav NR	Kravbeskrivning
1	Appen måste inkludera en funktion för visuell representation av addition.
2	Appen måste inkludera en funktion för visuell representation av subtraktion.
3	Appen måste inkludera en funktion för visuell representation av multiplikation.
4	Appen måste innehålla en funktion för visuell representation av division.
5	Appen ska erbjuda en funktion för visuell representation av användarens förbättring över tid.
6	Appen måste vara kompatibel med minst två olika modeller av iOS-mobiler.
7	Appen måste vara kompatibel med minst två olika modeller av Android-mobiler.
8	Appen måste vara kompatibel och fungera på minst en iPad-enhet.
9	Appen måste vara kompatibel och fungera på minst en Android-surfplatta.
10	Appen måste ha funktionalitet för att kommunicera med en databas.
11	Appen ska ha möjlighet att spara data lokalt på enheten under offlineanvändning // och sedan synkronisera och skicka den sparade datan till databasen när internetuppkopplingen återupprättas.
12	Appen ska spara versionen av applikationen i databasen för att möjliggöra jämförelse av statistik från olika versioner.
13	Information om skola och klass som skrivs in av lärare i appen ska sparas i databasen för att möjliggöra beräkning av användarstatistik.
14	Appen måste vara användbar både med och utan tillgång till internet.
15	Appen måste ha en svarstid på högst 150 millisekunder från det att ett svar matas in tills en ny fråga visas på en iPhone 5S.
16	Appen måste vara kompatibel och kunna köras på den fysiska iPhone 5S som tillhandahålls av kunden.
17	Appen ska inte innehålla någon text förutom namn, inmatning av klass och skola.
18	Eventuella arkitekturförändringar på databasen samt ändringar på den befintliga AI-algoritmen ska diskuteras med kunden innan de genomförs.



4.2 Nya krav från kunden

Tabell 2: Tabell över nya krav från kunden.

Krav NR	Kravbeskrivning	Krav NR i kravspec
19	Applikationen ska synkronisera data mellan enheter så att användare kan använda applikationen på flera enheter. Ej samtidigt.	24
20	Applikationen ska stödja flera användare på samma enhet.	20
21	Applikationen ska stödja att flera användare registreras med samma e-mail.	21
22	Applikationen ska autentisera skapade konton via e-mail.	22
23	Applikationen ska lagra statistik i databasen för hur länge användare befinner sig på visualiseringssidan.	2
24	Applikationen ska autentisera inloggning från en ny enhet via e-mail.	23
25	Applikationen ska kunna hämta ut uppgifter från optimeringsalgoritmen.	1
26	Applikationen ska spara data lokalt om internetuppkoppling inte finns tillgängligt. Denna data ska sedan skickas till databasen så fort internetuppkoppling är tillgänglig.	7
27	All funktionalitet i applikationen ska fungera på Android 5 (API Level 21) eller senare.	25
28	All funktionalitet i applikationen ska fungera på iOS 12 eller senare.	26
29	Det ska gå att använda applikationen för att räkna matematik utan tillgång till internet.	30
30	Användaren ska kunna skapa en profil i applikationen utan tillgång till internet.	31
31	Applikationen ska utvecklas i Flutter.	32



5 VAL, BEGRÄNSNINGAR OCH MOTIVERINGAR

I de två ursprungliga versionerna av Optimalt Lärande utvecklades en app med en funktionalitet som tillät användare att öva på de fyra räknesätten genom att lösa givna räkneuppgifter. En algoritm används för att välja uppgifter i en optimal ordning. I den kommande versionen kommer användargränssnittet att utökas för att utveckla animerade och interaktiva visualiseringar av de fyra räknesätten. Dessutom kommer progressionsskärmen vidareutvecklas för att visa användarens framsteg enligt den beräknade nivån.

Det finns flera begränsningar, några av de identifierade begränsningarna inkluderar att den befintliga applikationen kan ha en påverkan på utvecklingen, speciellt med tanke på tidigare beslut som redan har fattats. Dessutom har projektgruppen haft det svårt med att exakt veta vilka krav och funktioner som behöver implementeras, vilket gör att projektet blir svårare att konkret strukturera upp. Applikationen är också avsedd att vara tillgänglig för olika smartphones, både nya och äldre modeller, vilket innebär att den inte får bli för krävande för prestanda. Detta kommer dessutom att limitera vilken typ av Android version eller rättare sagt vilken Android API-nivå som kommer behövas ta hänsyn till för att få med så gamla mobiler som möjligt. Dessutom har gruppen en begränsad tidsram för vidareutvecklingen av applikationen, vilket betyder att den valda arkitekturen måste vara genomförbar inom denna tidsbudget.

En annan utmaning är främst bristen på förståelse när det gäller den befintliga applikationen, strukturen samt programmeringsspråket Flutter. Detta är en påverkande faktor som begränsar utvecklingen av projektet eftersom gruppmedlemmarna måste sätta sig in i att förstå kod som inte alltid är väldokumenterade av tidigare PUM-grupper.

Arkitekturen för den första versionen inkluderade främst delsystemen, databas, webserver och front-end-applikation. Frontenden skickar information till databasen utan att begära information tillbaka, vilket begränsar möjligheten att dela profiler på olika konton. Applikationen kräver inte en kontinuerlig anslutning mellan backend och frontend, eftersom backend-webbservern anropas när uppkopplingen återställdes igen. Applikationen ska ha möjligheten att dela ett konto på flera profiler dvs på flera enheter genom att kunna skapa synkade profiler med varandra genom ett konto. Dessutom ska det även vara möjligt att skapa profiler, spela spelet och skapa ett konto/profil utan en mejladress.

Beslut för den minimala produkten innefattar att behålla den övergripande arkitekturen från de tidigare versionerna. Fokus kommer att ligga på att utöka visualiseringar för de fyra räknesätten, inklusive animeringar och interaktivitet, samt att lägga till en skärm som visar progressionen. Databasen kommer att utökas med information om klass och skola kopplad till en profil. I helhet en förbättring visuellt med några extra tillägg.

Framtida utökningar som diskuterades men inte ingår i den minimala produkten inkluderar en språkförändring, vilket ska göra det möjligt för användaren att välja språk när hen skapar en profil. Detta är lågt prio då målet är att skapa ett spel som använder sig så lite som möjligt av ord och mer av siffror samt symboler.



6 ARKITEKTONISKA MEKANISMER

- Kommunikationsmönster: Beskriver hur olika delar av systemet kommunicerar med varandra, inklusive klient-server-kommunikation och gränssnitt hantering. För närvarande används en enkel klient-server-modell för kommunikation mellan front-end och back-end. Gränssnittet är statiskt och behöver utvecklas för att bli mer interaktivt.
- Datalagring mekanism, beskriver huvudsakligen hur data lagras och hanteras i systemet, detta omfattar även databasstruktur samt datalagring principer. För närvarande används en databas för att lagra konton samt profiler och övningsdata. Databas strukturen är enkel men behöver utökas för att kunna stödja nya krav från kunden.
- Användargränssnitt hantering, specificerar hur användargränssnittet interagerar med användaren och presenterar den visuella informationen på ett användarvänligt sätt. I projektets fall kommer användaren oftast vara barn. För närvarande begränsas användargränssnittet till att visa enkla visualiseringar och räkneuppgifter, något kunden vill att projektgruppen ska utveckla. Utökningen ska inkludera interaktiva och visuella element för att förbättra användarupplevelsen.
- Beteendemönster, definierar generellt hur systemet beter sig i olika situationer och hur systemet hanterar olika händelser och processer. För närvarande finns det endast grundläggande beteendemönster för hantering av användarinteraktion och dataflöde. Detta ska utvecklas och förbättras för att hantera mer komplexa användarscenarier och systemhändelser.

7 NYCKELABSTRAKTIONER

- Profil
 - Syftet är att spara optimeringsdata, statistik och progression i appen.
 - Det ska vara möjligt att spara profilen lokalt på enheten och ska kunna användas online samt offline.
 - Det ska vara helt anonymt (profilnamnet behöver inte skickas till databasen), med tanke på GDPR.
- Konto
 - Syftet med kontot är att synka profiler mellan enheter utan att behöva skapa en profil med en mejladress.
 - All data om kontot ska sparas i databasen och kan bara användas online.
- Klass
 - Syftet med klass är att länka samman konton/profiler så att läraren kan sammanställa och överblicka statistik.
 - Klassen ska sparas i databasen.
 - Administreras av kunden som sker huvudsakligen utanför appen.



8 ARKITEKTONISKA DESIGNMÖNSTER

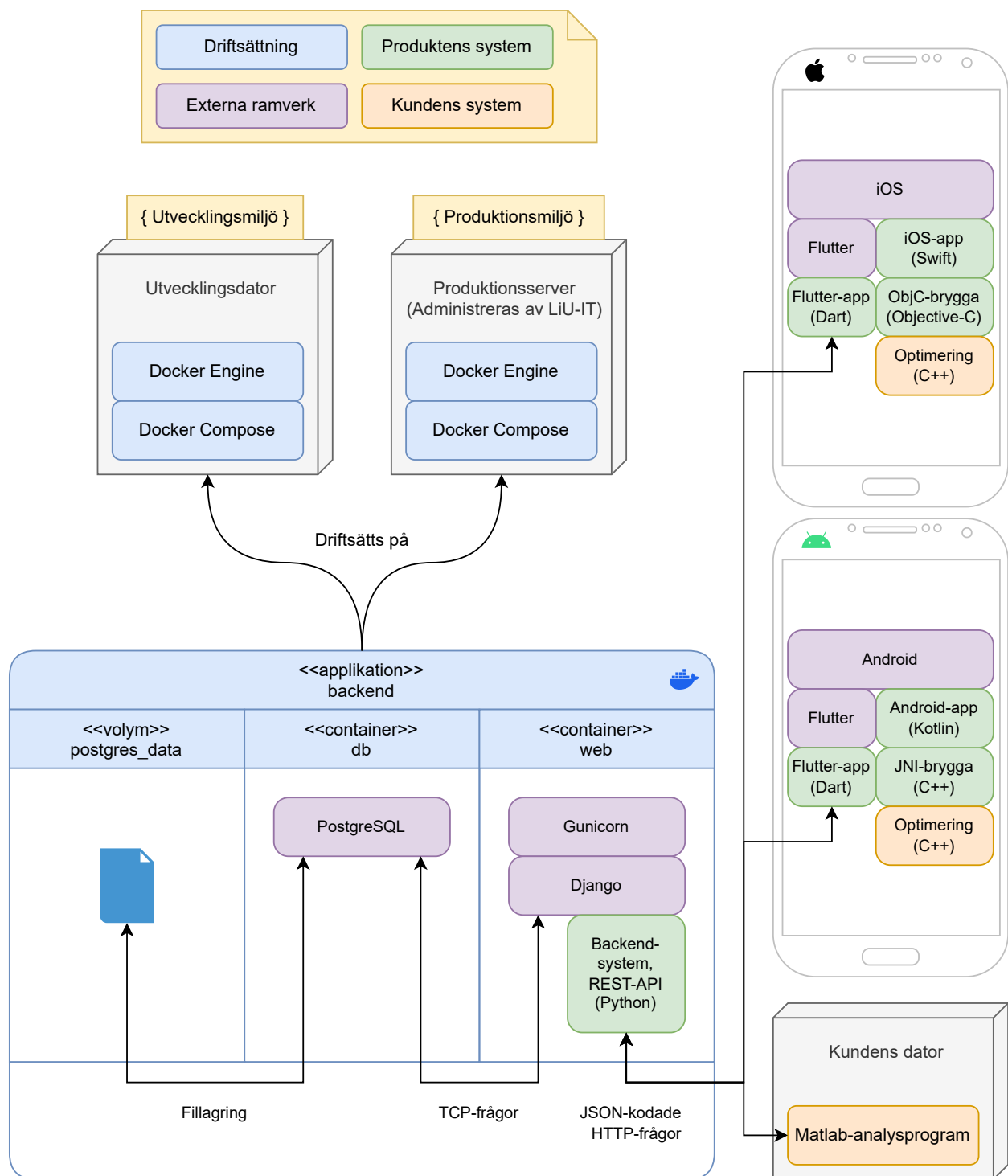
Tabell 3: Tabell över arkitektoniska designmönster

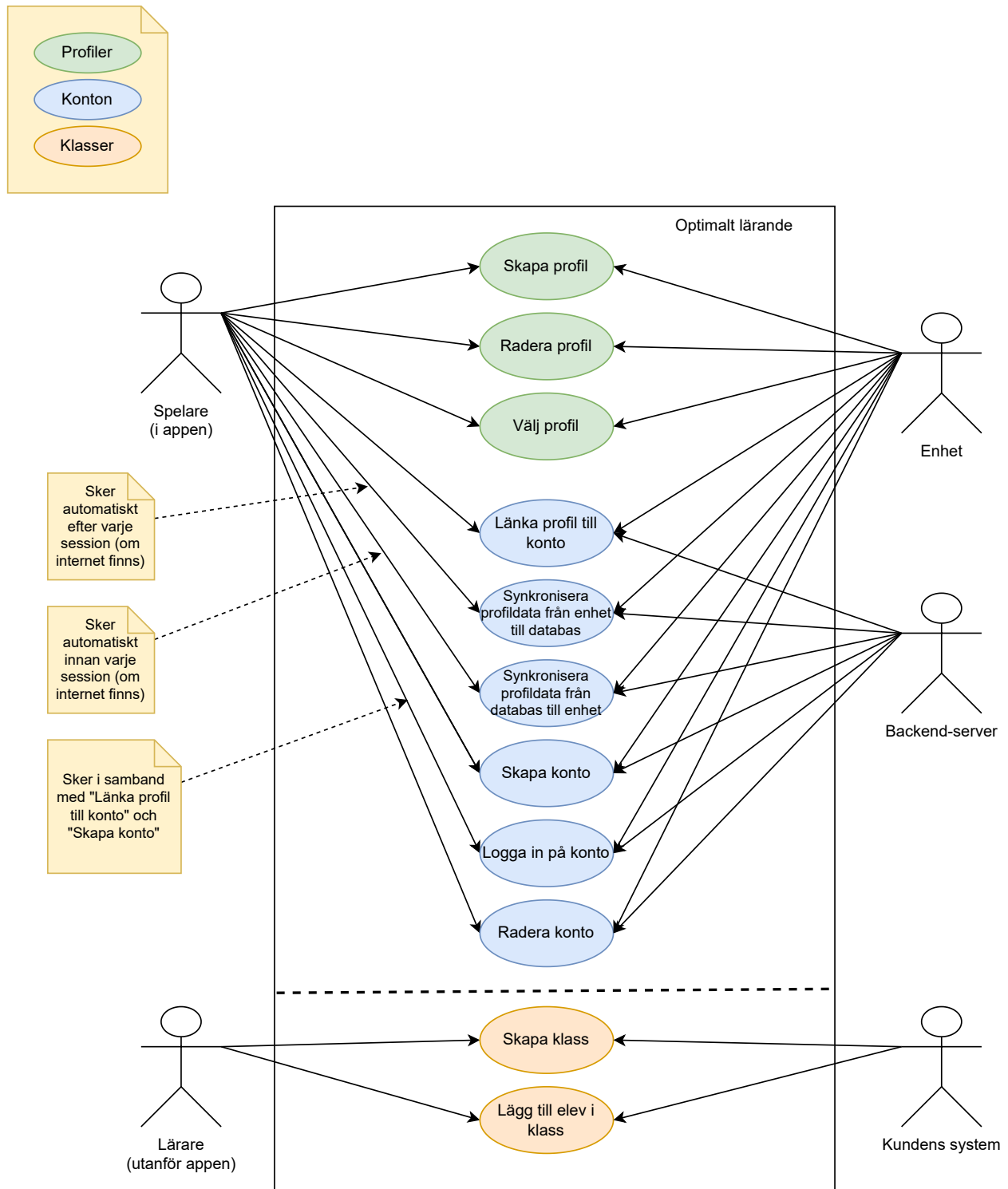
Designmönster	Delsystem	beskrivning
Model-View Controller (MVC)	Backend	Designmönstret består av tre komponenter som samarbetar för att skapa en fungerande applikation. - Modell: Hanterar de grundläggande datastrukturerna som används i databasen. Dessa inkluderar modeller för användare, enheter (en för Android och en för iOS), användarsessioner, användaruppgifter och knapptryck. - Vy: Ansvarig för att leverera svar till mobilapplikationen. Det finns olika vyer för att skapa användare, spara träningssessioner (statistik) och visa antalet användaruppgifter. - Styrenhet (Controller): Ansvarig för att uppdatera både vy och modell. Den styrs genom REST API:et för att hantera kommunikationen mellan applikationen och dess bakre del.
REST API	Backend	Applikationsgränssnittet (API) fungerar som en intermediär som styr kommunikationen mellan webbservern och mobilapplikationen. Gränssnittet är utformat enligt REST-designmönstret, detta innebär att en mobilapplikation kan interagera med gränssnittet genom att göra enkla requests istället för att behöva underhålla en pågående session.
Vyhierarki	Frontend	Användarens visuella upplevelse är baserad på en stack struktur där varje vy representerar en sida i applikationen. Den översta vyn i stacken är den som för närvarande visas för användaren. När användaren navigerar till en ny sida läggs en ny vy till i stacken, och när användaren backar tas den aktuella vyn bort från stacken för att visa den tidigare vyn som då hamnar överst i stacken.
Offline-first	Frontend Backend	Offline-first är en arkitektonisk designprincip som kan tillämpas på både backend och frontend av en applikation. Det handlar om att prioritera funktionalitet som gör att användare kan interagera med applikationen även när de inte har tillgång till internet. Således kan offline-first-principen implementeras både i frontend för att hantera gränssnittet och i backend för att hantera dataåtkomst och synkronisering med en server. Det är alltså inte specifikt en backend- eller frontend-teknik, utan snarare en övergripande strategi för att utforma applikationer med offline-funktionalitet i åtanke.



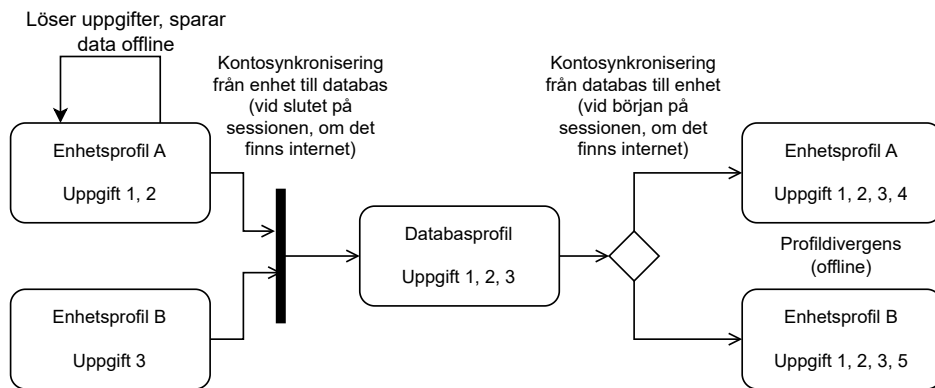
9 ARKITEKTONISKA VYER

Figur 1 visar sambandet och länken mellan mobilen där appen kommer finnas och självaste backend och systemet. Denna länkning sker via både en IOS och en Android där färgerna visar vilket lager varje modul är på i både mobil mässigt men även i systemet. Figur 2 är ett use case diagram som tydligt visar vilka aktörer som kommer att göra dessa use cases. Backend-servern som innehåller databasen kommer hantera flera use cases såsom att synkronisera profiler men det är viktigt att urskilja backend-servern med enheten då data kommer att behövas sparas lokalt för att kunna skapa konton offline osv. Aktören lärare är ett krav från kunden som ska ha tillgång till att skapa klasser av elever som hen sedan kan se över via statistik, topplistor, med mera. Figur 3, representerar ett flödesschema över länken mellan profiler och konton. Ett konto kan ha flera profiler, men en profil kan inte ha flera konton. Det är viktigt att urskilja dessa två begrepp då tanken är att man ska kunna synkronisera sitt konto från olika enheter där varje enhet har sin specifika profil. För att skapa en helhetsbild av systemet har en systemanatomi tagits fram, se figur 4.

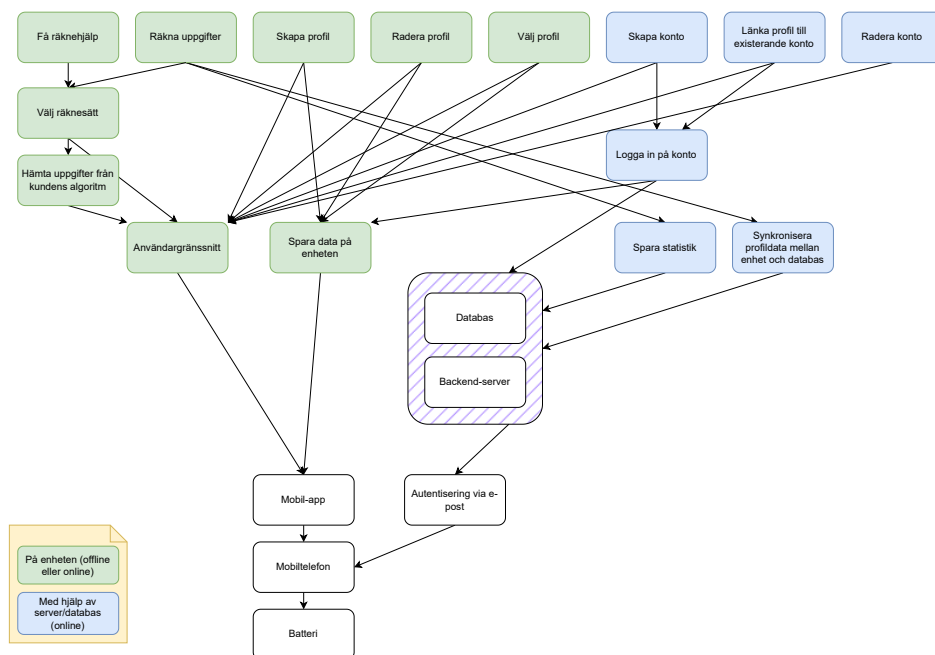
**Figur 1:** Hierarkin mellan mobiltelefoner och backend



Figur 2: Use case diagram



Figur 3: Flödesschema för synkning av profil och konto



Figur 4: Systemanatom



10 REFERENSER

[1] PUM09, “Kravspecifikation” 2024.