

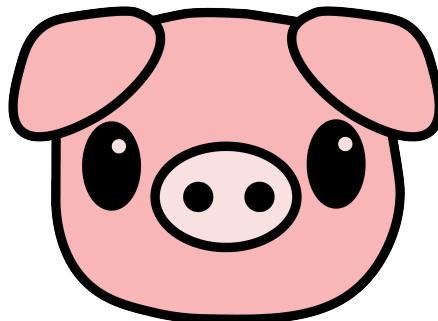
**INSTITUTT FOR DATATEKNOLOGI OG INFORMATIKK**

**IDATT2106 - SYSTEMUTVIKLING 2**

---

**Sparesti**  
**Hovedrapport**

---



*Forfattere:*

Elias Trana  
Emil Skogheim  
Erik Nordsæther  
Lars Talian Stangebye-Hansen  
Sander Rom Skofsrud  
Vegard Johnsen  
Kristoffer Fredriksen  
Jacob Forsdahl Iqbal

Mai, 2024

---

## Sammendrag

Gjennom denne rapporten dokumenteres utviklingen til Team 5 webapplikasjon, utviklet av Dataingeniør-studenter ved Norges teknisk-naturvitenskaplige universitet (NTNU) i samarbeid med produkteire fra Digital Transformasjon. Applikasjonen har som hovedmål å motivere voksne til å forbedre sin økonomi tilpassede spareutfordringer og individuelle tips.

Personlig økonomi er et felt som har en kontinuerlig store interessegruppe som alltid er på utkikk etter økonomiske verktøy og faglige tips. Ettersom feltet berører så mange blir også en betydelig mengde forskning- og næringslivsinvestering gjort for å fremme positive økonomiske vaner. Sentralt i prosjektet står en integrasjon av sosiale aspekter, som vil fungere som verktøy for å fremme samarbeid og interaksjoner rundt personlig økonomi. En sosial feed gjør at brukere kan dele mål og utfordringer med venner, samt se og interagere på venners aktivitet.

Mål og utfordringer kan opprettes personlig av brukeren, samt genereres ved hjelp av kunstig intelligens eller velges fra en forhåndsdefinert liste med populære maler. Målene kan også deles med venner slik at en samarbeidskomponent blir integrert i applikasjonen. Brukeren blir også tildelt streaks og medaljer for å belønne dens innsats.

Summen av disse aspektene i denne unike applikasjonen resulterer i en brukervennlig og motiverende plattform som vil fungere som et verktøy for å motivere voksne til å forbedre deres økonomi på en sosial og engasjerende måte.

---

## Forord

Rapporten er skrevet som en del av emnet IDATT2106 Systemutvikling 2 ved Norges teknisk-naturvitenskaplige universitet (NTNU). Oppgaven ble valgt grunnet et samlet ønske om å skape en løsning som kan hjelpe voksne med å forbedre økonomien sin på en funksjonell og engasjerende måte. Prosjektet ble gjennomført over en periode på 3 uker. Dette gjorde at prosessen frem til resultatet har vært til tider krevende, men også svært lærerikt for alle teammedlemmene. Gjennom to sprinter har teamet brukt smidig metodikk for å utvikle og forbedre applikasjonen. Dette ble gjort i samspill med jevnlige tilbakemeldinger fra produkteiere og brukertesting for å sikre kvalitet og brukervennlighet.

Teamet ønsker å takke Grethe Sandstrak, Surya Kathayat og Muhammad Ali Norozi for deres uvurderlige veiledning, tilbakemelding og støtte gjennom prosjektet. En spesiell takk rettes også til Pedro Pablo Cardona Arroyave for hans hjelp underveis, både når det kom til generelle utviklingsspørsmål, samt flere utbytterike brukertester. Til slutt går takken til produkteierne, Thea Marie Semundset Løseth og Jule Bue Kulseth for et engasjerende samarbeid bestående av betydningsfull innsikt og faglig kompetanse.



Elias Trana



Vegard Johnsen



Sander Rom Skofsrud



Erik Nordsæther



Jacob Forsdahl Iqbal



Kristoffer Fredriksen



Emil Skogheim



Lars Talian Stangebye-Hansen

Trondheim, 15. mai 2024

# Innhold

<b>Figurer</b>	<b>iv</b>
<b>Tabeller</b>	<b>v</b>
<b>1 Introduksjon</b>	<b>1</b>
1.1 Definisjoner, akronymer og forkortelser . . . . .	2
<b>2 Teori og relevant litteratur</b>	<b>3</b>
2.1 Tidligere kunnskaper og erfaringer . . . . .	3
2.2 Scrum-prosessen . . . . .	3
2.2.1 Artefakter i Scrum . . . . .	3
2.2.2 Burndown . . . . .	4
2.3 GDPR og personvern . . . . .	4
2.4 Programvareutvikling . . . . .	5
2.4.1 Støtteverktøy . . . . .	5
2.4.2 UX - Bruker opplevelse . . . . .	5
2.4.3 Modellering . . . . .	5
2.4.4 Systemarkitektur . . . . .	6
2.4.5 Lagring . . . . .	6
2.4.6 Sikkerhet . . . . .	6
2.4.7 Testing . . . . .	7
2.4.8 Monorepo . . . . .	7
<b>3 Metode</b>	<b>8</b>
3.1 Produkteinere . . . . .	8
3.2 Artefakter . . . . .	8
3.3 Kravidentifisering . . . . .	8
3.3.1 Designprinsipp . . . . .	8
3.3.2 Datainnsamlingsmetode . . . . .	8
3.4 Scrum . . . . .	9
3.4.1 Sprintrutine . . . . .	9
3.4.2 Sprintplanlegging . . . . .	9
3.4.3 Standup . . . . .	10
3.4.4 Burndown . . . . .	10
3.4.5 Sprint retrospektiv . . . . .	10
3.5 Rammeverk, APIer og biblioteker . . . . .	11
3.5.1 Klientside . . . . .	11
3.5.2 Tjenerside . . . . .	11
3.6 CI/CD . . . . .	11
3.7 Dokumentasjon . . . . .	11
3.8 Byggeverktøy . . . . .	11
3.9 Utviklingsverktøy . . . . .	12
3.10 Versjonskontroll . . . . .	12
3.11 Arbeidsfordeling . . . . .	12
3.12 Sikkerhet . . . . .	12
3.13 Modellering og design . . . . .	13
3.13.1 Prototyper . . . . .	13
3.13.2 API Design . . . . .	13
3.14 Testing . . . . .	13
3.14.1 Klientside . . . . .	13
3.14.2 Tjenersiden . . . . .	13
<b>4 Resultater</b>	<b>14</b>
4.1 Bemerkning . . . . .	14
4.2 Prosjektresultater . . . . .	14
4.2.1 Akseptansekriterier og oppnåelse . . . . .	14
4.2.2 Tilbakemelding fra produkteinere (Sprint 1): . . . . .	14

4.2.3	Tilbakemelding fra produkteiere (Sprint 2):	15
4.2.4	Produkteiers prioriteringer:	15
4.2.5	Brukertester	15
4.2.6	Systemarkitektur	15
4.2.7	Testing	16
4.2.8	Samarbeid med tredjeparts API	16
4.3	Administrative resultater	17
4.3.1	Digital deltaker	17
4.3.2	Teamutvikling	17
4.3.3	Utførte sprintretrospektiver	17
4.3.4	Sprintkøen	17
4.3.5	Burndown-diagram	18
4.3.6	Bussfaktor	18
<b>5</b>	<b>Konklusjon og videre arbeid</b>	<b>19</b>
5.1	Videre arbeid	19
5.1.1	Ferdigstille tekniske krav	19
5.1.2	GDPR og personvern	19
5.1.3	Videre utvikling av tredjeparts API	19
5.1.4	Fremtidige implementasjoner	19
5.1.5	Videre prosessarbeid	20
5.1.6	Drøfting	20
5.2	Konklusjon	20
	<b>Referanser</b>	<b>21</b>
	<b>Vedlegg</b>	<b>22</b>
	Visjonsdokument	A1
	GitLab lenker	B1
	Møteinkalling fra sprint 1 planlegging	C1
	Møtereferat fra sprint 1 planlegging	D1
	Møteinkalling for møte med produkteier 19.04.2024	E1
	Møtereferat for møte med produkteier 19.04.2024	F1
	Møteinkalling for sprint 1 review 25.04.2024	G1
	Møtereferat for review 1 25.04.2024	H1
	Møteinkalling for sprint 2 planlegging 25.04.2024	I1
	Møtereferat for sprint 2 planlegging 25.04.2024	J1
	Møteinkalling for sprint 2 review 03.05.2024	K1
	Møtereferat for sprint 2 review 03.05.2024	L1
	Produkteiers prioriteringer	M1
	Wireframe	N1
	Visuell profil	O1
	Usertest 1	P1
	Usertest 2	Q1
	Produktkø	R1
	Sprintkø Sprint 1	S1
	Sprintkø Sprint 2	T1
	ER diagram fra 25.04	U1
	Domenemodell	V1
	Wiki	W1

## Figurer

1	Sprint 2 issueboard, dag 1	9
2	Burndown sprint 1	10
3	Burndown sprint 2	10
4	Systemarkitektur	16
5	Wireframe	N1

6	Wireframe - Sosial . . . . .	N2
7	Wireframe - Mål . . . . .	N2
8	Wireframe - Konfigurering . . . . .	N3
9	Endelig design - Forsiden av applikasjonen . . . . .	O1
10	Endelig design - Applikasjonens egenskaper . . . . .	O1
11	Endelig design - Logg inn visningen . . . . .	O2

## Tabeller

1	User story oppnåelser (Sprint 1) . . . . .	14
2	User Story oppnåelser (Sprint 2) . . . . .	15

---

# Oppgavebeskrivelse

Opprinnelig var oppgaven å utvikle webapplikasjonen SpareSti, som skulle motivere voksne til å forbedre økonomien sin gjennom personlig tilpassede sparetips og budsjettverktøy. Prosjektet skulle integrere løsningen med nettbanker for å gi brukerne oversikt over økonomien og muligheter til å sette av penger. Underveis utviklet oppgaven seg basert på ønsker og innspill fra produkteierne, som fokuserte mer på det sosiale aspektet av applikasjonen og reduserte vektleggingen på budsjettering og nettbank integrering.

## 1 Introduksjon

Denne rapporten er skrevet som den avsluttende delen av prosjektet i emnet IDATT2106 Systemutvikling 2 med smidig prosjekt. Prosjektet har hatt varighet fra mandag 15. april til fredag 3. mai. Hovedformålet med oppgaven var å utvikle en applikasjon for å fremme sparing hos voksne ved bruk av sparemål og utfordringer. Mange synes det kan være vanskelig å komme i gang med sparing, både grunnet mangel på oversikt over egen økonomi, manglende motivasjon samt manglende kunskap og kompetanse rundt sparing. Programmet er utviklet med formål å dekke disse behovene.

Hovedinteressentene for prosjektet var produkteierne som ønsket en applikasjon for å dekke nettopp disse behovene. I dette prosjektet er produkteierne masterstudenter ved Digital Transformasjon ved NTNU.

Rapporten er strukturert med en teori-del, metode, resultater og konklusjon. Teorien inneholder relevant teori, både rundt Scrum i tillegg til teori rundt teknologier. Metode inneholder informasjon om hvordan gruppen har arbeidet i prosjektet. Her diskuteres både Scrum i tillegg til teknologier, testing og sikkerhet. Resultatdelen inneholder en drøfting av Teamets resultater opp mot akseptansekriteriene stilt av produkteierne, og i konklusjonen blir det drøftet videre arbeid for hele rapporten avsluttes med en konklusjon av vårt arbeid.

---

## 1.1 Definisjoner, akronymer og forkortelser

- API: Application Programming Interface
- CORS: Cross-Origin Resource Sharing
- Commit: En handling som registerer en endring i kildekoden opp mot versjonskontrollsystemet.
- CSRF: Cross-Site Request Forgery
- Domene modell: Konseptuell modell av sammensetningen til dataen i et system.
- E2E: End-to-end
- ER-modell: Entity Relasjons-modell, brukes for å visuelt representer tabeller i en database og deres sammenhenger.
- IDATT1002: Emnet Systemutvikling 1.
- IDATT2105: Emnet Fullstack Applikasjonsutvikling.
- IDATT2106: Emnet Systemutvikling 2 med smidig prosjekt.
- Figma: Program brukt for å lage design prototyper og wireframes.
- HTTP: Hypertext Transfer Protocol
- Hibernate: En ORM for Java.
- JSON: JavaScript Object Notation
- JPA: Java Persistence API
- JWT: JSON Web Token
- Mocking: Forskjellige teknikker for å fastsette oppførselen til et objekt eller en funksjon.
- Merge: En handling der endringer i en branch blir integrert med endringer i en annen branch.
- MVP: Minimal Viable Product
- ORM: Object Relational Mapper
- SQL: Structured Query Language
- OWASP: The Open Worldwide Application Security Project
- REST: Representational State Transfer.
- Scrum: En smidig metodikk som følger en iterativ prosess med faste seremonier og artefakter.
- Sprint: En arbeidsperiode med satte oppgaver som pleier å vare mellom 1-3 uker.
- UX: User Experience
- Versjonskontrollsyste: Programvare som muliggjør endringer i kildekode, samt samarbeid, og gjenopprettning av tidligere versjoner.
- Vite: Lokal utviklingsserver for klientsiden.
- WCAG: Web Content Accessibility Guidelines
- NPM: Pakkebehandler for JavaScript prosjekter.

---

## 2 Teori og relevant litteratur

I denne seksjonen presenteres det teoretiske grunnlaget og metodene som støtter vårt ingeniørfaglige arbeid. Det vil dekke viktige områder som smidig programvareutvikling og universell utforming sammen med andre relevante teknologiske og vitenskapelige prinsipper. Formålet er å gi leseren nødvendig innsikt for å forstå og evaluere løsningene som er utviklet.

### 2.1 Tidlige kunnskaper og erfaringer

I faget IDATT1002 opparbeidet teammedlemmene grunnleggende kunnskaper om systemutviklingsprosesser og teamarbeid. De lærte også om utviklingen av produkter for en klient, samt hvordan man skriver rapporter og dokumenterer tekniske systemer. I tillegg har teammedlemmene erfaring fra tidlige øvinger og prosjekter i faget IDATT2105, hvor de ble introdusert for samspillet mellom klient og server ved hjelp av REST-rammeverk, og utvikling av programmer som interagerer med databaser.

### 2.2 Scrum-prosessen

Scrum-prosessen er et smidig rammeverk som støtter iterativ utvikling gjennom definerte artefakter og strukturerte seremonier. Denne tilnærmingen fremmer samarbeid og kontinuerlig forbedring, med fokus på korte sprints for å oppnå raske og effektive leveranser. De følgende avsnittene tar for seg nøkkelkomponentene i Scrum, inkludert artefakter og seremonier, som bidrar til å sikre transparens, forutsigbarhet og tilpasning gjennom hele prosjektet.

#### 2.2.1 Artefakter i Scrum

I Scrum er artefaktene viktige fordi de bidrar til å danne en felles forståelse for alle involverte. Disse artefaktene inkluderer produktbackloggen, sprintbackloggen, og brukerhistorier, og hver spiller en kritisk rolle i å styre både planleggingen og utførelsen av arbeidet i Scrum-prosjekter.

#### Produkt-backlog

Produktbackloggen er en kontinuerlig oppdatert liste som omfatter alt som er nødvendig for å forbedre et produkt. Denne listen er den eneste kilden til arbeidsoppgaver for Scrum-teamet. I sprintplanleggingen velges og raffineres oppgaver fra produktbackloggen for å sikre at de er klare og gjennomførbare (Scrum.org 2024b).

#### User Stories

User stories er en sentral del av produktbackloggen og er en beskrivende fremstilling av en programvarefunksjon fra en brukers perspektiv. Hovedformålet er å fange opp et spesifikt brukerbehov som en funksjonell del av arbeidet. "Som [brukertype], ønsker jeg [funksjon], så jeg kan [oppnå et mål]. Hver user story skal bidra til produktets verdi, uavhengig av implementeringsrekkefølgen (Alliance 2024).

#### Sprintbackloggen

Sprintbackloggen består av sprintmålet (hvorfor), de utvalgte produkt-backlog-elementene for sprinten (hva) og en konkret plan for utførelsen (hvordan). Dette dokumentet fungerer som en plan laget av og for utviklere og gir et tydelig og oppdatert bilde av arbeidet de planlegger å gjennomføre under sprinten for å nå sprintmålet. Sprintbackloggen oppdateres kontinuerlig gjennom sprinten

---

etter behov og inneholder tilstrekkelig detaljer slik at utviklere kan vurdere fremgangen under det daglige Scrum-møtet. Sprintmålet, som defineres under sprintplanleggingen og legges til i sprint-backloggen, fungerer som sprintens enkelte mål (Scrum.org 2024c).

### **2.2.2 Burndown**

Et burndowndiagram er et visuelt verktøy, brukt for å spore gjenværende arbeid i et prosjekt eller en sprint. Diagrammet viser mengden av estimert gjenværende arbeid som reduseres over tid, og hjelper teamet med å vurdere om de er på vei til å fullføre arbeidet innen tidsfristen. Dette gir innsikt i prosjektets fremgang og kan avdekke behov for justeringer hvis fremgangen avviker fra planen.

### **Sprintplanlegging**

Sprintplanlegging er en viktig seremoni i Scrum der teamet samles for å bestemme hvilke oppgaver fra produktbackloggen som skal tas med i den kommende sprinten. Under denne seremonien definerer teamet også sprintmålet, som gir en klar retning for arbeidet som skal utføres. Dette møtet er viktig for å sikre at alle teammedlemmene har en felles forståelse for oppgavene og målene for sprinten, samt at det legger grunnlaget for arbeidet som skal gjøres. Sprintplanleggingen innebærer også utarbeidelse av en detaljert plan for hvordan oppgavene skal gjennomføres, noe som reflekteres i sprintbackloggen. Denne prosessen er avgjørende for å koordinere innsatsen og sikre at teamet er godt forberedt til å møte sprintens utfordringer.

### **Standup**

Standup er et kort, møte hvor Scrum-teamet vurderer fremgangen mot sprintmålet og tilpasser sprintbackloggen etter behov. Møtet er designet for å styrke kommunikasjon, identifisere hindringer, og fremme rask beslutningstaking. Det holdes på samme tid og sted hver dag og involverer primært utviklerne, selv om produktansvarlig og Scrum Master kan delta hvis de aktivt arbeider med oppgaver. Scrum Master sikrer at møtet er effektivt og holder seg innenfor tidsrammen (Scrum.org 2024a).

### **Retrospektiv**

Retrospektiv er en seremoni i Scrum hvor teamet reflekterer over den siste sprinten for å øke kvaliteten og effektiviteten i fremtidige sprinter. Under dette møtet vurderer teamet hva som fungerte godt, hvilke utfordringer de møtte, og hvordan disse utfordringene ble håndtert. Det blir også diskutert forbedringer i arbeidsprosessene. Seremonien avslutter hver sprint og resulterer i konkrete tiltak som teamet vil implementere i den neste sprinten. Dette gjør retrospektiven til en viktig mulighet for teamet å fokusere på evaluering og tilpasning.

## **2.3 GDPR og personvern**

GDPR er et regelverk designet for å beskytte persondata og privatlivet til brukerne. Det gir individer retten til å få tilgang til deres personlige persondata som behandles av organisasjoner, og å be om sletting av disse dataene om ønskelig. GDPR gjelder for alle organisasjoner i EU og EØS som behandler personopplysninger. I tilfeller der et produkt kun brukes lokalt i en akademisk kontekst, vil ikke GDPR-reglene være gjeldende lenger (Datatilsynet 2023).

---

## 2.4 Programvareutvikling

Denne delen beskriver de tekniske metodene og verktøyene som ble benyttet i utviklingen av prosjektet. Det har vært fokusert på å implementere effektive og moderne programvareutviklingspraksiser som versjonskontroll, byggeverktøy, og CI/CD prosesser.

### 2.4.1 Støtteverktøy

#### Versjonskontroll

Versjonskontroll er en metode brukt i programvareutvikling for å systematisk spore endringer i kildekoden over tid. Denne metoden støtter en smidig og organisert utviklingsprosess ved å muliggjøre arbeid på samme kodebase samtidig. I de fleste versjonskontrollsystemer håndteres parallelt arbeid gjennom bruk av branches. Hver branch representerer en egen versjon av kodebasen, og kan modifiseres uavhengig av hovedgrenen (main branchen). Endringer i en branch kan deretter lett integreres tilbake til "main branchen" uten å forstyrre arbeidet på andre branches, noe som tilbyr en fleksibel og oversiktlig utviklingsprosess.

#### Byggeverktøy

Byggeverktøy automatiserer prosessen med å kompilere programvare ved å håndtere avhengigheter og koordinere byggeprosessen gjennom en konfigurasjonsfil. Denne filen definerer rekkefølgen komponentene skal bygges i og hvilke verktøy som skal brukes for å kompilere og lenke kildekoden. Ved å sikre konsistente og repeterbare byggeprosesser på tvers av ulike miljøer, sparer byggeverktøyet tid og reduserer feil ved å automatisk oppdatere alle berørte komponenter basert på endringer i kildekoden eller avhengige biblioteker.

#### CI/CD

Continuous Integration (CI) og Continuous Delivery/Deployment (CD) er kjernekomponenter i moderne programvareutvikling, som øker kodekvalitet og forbedrer samarbeid. CI innebærer at utviklere regelmessig "commiter" sin kode til et versjonskontrollsysteem. Dette gjør det mulig å oppdage feil tidlig ved å arbeide med mindre kodeendringer, noe som reduserer sannsynligheten for komplekse "merger". CD utvider CI ved å automatisere overføring av kode til produksjon etter vellykket testing, noe som sikrer at koden rulles ut raskt og pålitelig. I praksis benytter man automatiserte byggeprosesser hvor programvare og database pakkes, og testes automatisk (Bass 2022).

### 2.4.2 UX - Bruker opplevelse

Brukeropplevelsdesign fokuserer på å skape produkter som gir meningsfulle og givende opplevelser for brukerne. Prosessen omfatter design av merkevare, brukervennlighet og funksjonalitet gjennom hele produktets livssyklus. En sentral del av UX-design er å sikre en konsistent og intuitiv brukerflyt gjennom produktet, ofte referert til som rød tråd". Dette prinsippet hjelper til med å lede brukeren gjennom programmets ulike deler på en logisk og forutsigbar måte, noe som er avgjørende for produktets suksess (I. D. Foundation 2024).

### 2.4.3 Modellering

Modellering er en viktig del av planleggingsfasen og innebærer å skape en forenklet fremstilling av et system. Dette gjøres for å analysere og formidle informasjon om systemets funksjonalitet

---

og struktur, noe som bidrar til en felles forståelse blant utviklere. I denne fasen er Entitets-Relasjonsmodellen (ER-diagram) ofte benyttet for å strukturere databasen. ER-modellen skildrer entiteter med deres attributter og illustrerer relasjonene mellom disse entitetene.

#### **2.4.4 Systemarkitektur**

##### **Lagdelt arkitektur**

Lagdelt arkitektur er en strukturert modell for programvareutvikling som deler applikasjonen inn i separate lag, hvert med sitt eget ansvarsområde og funksjonalitet. Denne arkitekturen består ofte av tre hovedlag: presentasjonslaget, forretningslaget og datalaget. Presentasjonslaget håndterer brukergrensesnittet og er det det brukerne direkte kommuniserer med. Forretningslaget inneholder kjernen i applikasjonens forretningslogikk. Datalaget tar seg av all datahåndtering, inkludert kommunikasjon med databaser, og sørger for dataintegritet og lagring.

##### **REST**

REST er en arkitekturstil for nettverksapplikasjoner, med fokus på enkelhet og skalerbarhet. Arkitekturen anvender standardiserte HTTP-metoder som GET, POST, PUT og DELETE, noe som gir forutsigbarhet i kommunikasjonen mellom klient og server. Kjernen i REST er statelessness, der hver forespørsel inneholder all nødvendig informasjon for å kunne behandles uavhengig, uten behov for at serveren lagrer tidligere klienttilstander. Arkitekturen inkluderer også et lagdelt system der klienter ikke trenger direkte tilgang til serveren, og serveren kan tilby kjørbar kode til klienten ved behov. Denne modulære og fleksible tilnærmingen gjør REST spesielt egnet for utvikling av robuste og lett vedlikeholdbare webtjenester (IBM 2024).

#### **2.4.5 Lagring**

##### **Datahåndtering og lagring**

Datahåndtering og lagring er kritiske aspekter i systemutvikling, der effektiv håndtering av data sikrer integritet, tilgjengelighet og sikkerhet. Effektiv datahåndtering innebærer lagring av data i strukturerte tabeller som kan manipuleres ved hjelp av SQL. Dette språket muliggjør effektiv spørring, oppdatering og styring av data, noe som er avgjørende for webapplikasjoner som krever raske og sikre transaksjoner. Moderne databaseteknologier tilbyr skalerbarhet og fleksibilitet, støtter komplekse spøringer og sikrer data mot uautorisert tilgang. Dette understreker viktigheten av et robust databasehåndteringssystem for effektiv datahåndtering i alle typer systemer (Oracle 2020).

##### **Object-Relational Mapping**

ORM er programvare som fungerer som en bro mellom objektorientert programmering og relasjonsdatabaser, ved å oversette data mellom de to systemene. ORMer øker produktiviteten ved å eliminere behovet for tilpassede databasemanipulasjoner til språkets egne strukturer uten å kreve noen kunnskap om databasespørringsspråk (Ellingswood 2024).

#### **2.4.6 Sikkerhet**

Autentisering og autorisering er to grunnleggende sikkerhetsmekanismer som beskytter APIer ved å kontrollere tilgang til data. Autentisering bekrefter identiteten til en bruker eller en tjeneste som prøver å få tilgang til APIet. Dette sikrer at enheten er den den hevder å være. Når identiteten er bekreftet, bidrar autorisering til å bestemme hvilke data eller handlinger brukeren har tillatelse

---

til å se eller utføre. Sammen bidrar disse prosessene til å beskytte sensitive data mot uautorisert tilgang og sikre at kun de med nødvendige rettigheter kan utføre handlinger eller se informasjon, noe som opprettholder integriteten (Kong 2023).

#### **2.4.7 Testing**

##### **Enhetstesting**

Enhetstesting er en type testing som fokuserer på individuelle enheter eller komponenter i programvaren. Hensikten med enhetstester er å validere at hver enhet av programkoden fungerer etter hensikten (geeks 2024). For å isolere og teste individuelle komponenter, benyttes ofte mocking sammen med enhetstester for å simulere avhengigheter og sikre at testene blir uavhengig av eksterne faktorer. Ved å mocke avhengigheter, kan testene kjøre i et kontrollert miljø hvor man har full kontroll på inn- og forventet utdata. Dette gir en mer effektiv og målrettet testprosess, hvor hver test kjører isolert med forutsigbare resultater, noe som er essensielt for pålitelig programvareutvikling (geeks 2023).

##### **E2E-testing**

E2E er en omfattende testmetode som brukes for å teste større applikasjoner fra start til slutt i en prosess som etterligner brukerinteraksjoner. Metoden tester applikasjonens funksjonelle flyt og datahåndtering på tvers av moduler. E2E-testing sikrer at alle integrerte deler av applikasjonen fungerer som forventet i det faktiske produksjonsmiljøet. Dette er en kritisk del av kvalitetssikringen som bidrar til å øke påliteligheten og brukeropplevelsen til det ferdige produktet (Microsoft 2023).

#### **2.4.8 Monorepo**

Monorepo er en versjonsstyringskonfigurasjon hvor flere prosjekt lagres i et enkelt repositorium. Denne tilnærmingen fremmer godt samarbeid og bidrar til god oversikt over prosjektene. Bruk av monorepo tilbyr flere fordeler: Det forenkler gjenbruk av kode, effektiviserer endringer i koden, og standardiserer byggeprosessen. I et monorepo kan én enkelt handling endre flere prosjekter samtidig, noe som bidrar til en mer effektiv utviklingsprosess (Perforce 2023).

---

## 3 Metode

### 3.1 Produkteinere

Teamet samarbeidet med produkteinerne Julie Bue Kuset og Thea Marie Semundseth Løseth gjennom hele prosjektet.

### 3.2 Artefakter

I løpet av prosjektet utnyttet gruppen en rekke forskjellige artefakter for å forbedre kommunikasjon og effektivitet. For å gi et visuelt bilde av fremgangen ble det benyttet både et fysisk og et digitalt issueboard, sistnevnte i form av IDI GitLab. I tillegg ble det brukt klisterlapper for å visualisere issues og fremdrift. Effektiviteten og fremgangen til teamet ble overvåket gjennom burndowngrafer. Et whiteboard var sentralt for å tegne forslag og diskutere implementeringer. Tilstandssjekker ble regelmessig brukt for å måle og vurdere teammedlemmene velvære. Retrospektive undersøkelser ble også gjennomført etter hver sprint for å identifisere hva som fungerte bra og hva som kunne forbedres i de påfølgende sprintene.

### 3.3 Kravidentifisering

Gjennom de muntlige dialogene som ble utført med produkteinerne ble ønskede implementeringer og akseptansekriterier identifisert. Disse kriteriene ble både dokumentert i form av fysiske userstories med prioritetsnivå, samt en skriftlig beskrivelse av de ønskede implementasjonene. Denne skriftlige beskrivelsen ble sendt til produkteinerne og godkjent av dem. Prioriteringen ble utført basert på userstoriene gitt fra produkteinerne.

#### 3.3.1 Designprinsipp

Systemet ble designet basert på beskrivelsene fra produkteinerne. Deres ønske om et stilrent produkt med en personlighet som tilskir ryddighet, men gøy, ble tatt i betraktning under designingen av applikasjonen. Logoen ble redesignet da den føltes personlighetsløs, noe produkteinerne var enige i. Den ble redesignet til en enklere logo med et tydelig uttrykk. Dette designet ble brukt til å skape en gjennomgående interaktivitet i programmet. Helhetlig bidro logo-endringen til å styrke den totale visuelle identiteten til applikasjonen. Fokus bak designet er at det skal være tilgjengelig på mobile applikasjoner. De aller fleste mobile applikasjoner utnytter hamburgermenyen, men ved spesielle tilfeller finnes det mobile applikasjoner som ikke endrer navigasjonsbaren til en hamburgermeny. Derfor var det viktig at navigasjonsbaren var stor nok slik at den fortsatt var tilgjengelig for berøringsenheter.

Under designprosessen har teamet lagt stor vekt på å overholde WCAG, for å sikre at applikasjonen er tilgjengelig for alle brukere, inkludert de med funksjonsnedsettelse. Navigasjonsbaren ble designet med fokus på enkel betjening på berøringsenheter og tilgjengelighet. Det ble også implementert tiltak som loading animasjoner, tydelige feilmeldinger og kontraster i fargevalg for å forbedre brukeropplevelsen og gjøre applikasjonen tilgjengelig for et bredt spekter av brukere.

#### 3.3.2 Datainnsamlingsmetode

Det ble utført to brukertester, en ble utført en brukertest før MVP og den andre brukertesten før levering av det ferdige produktet. Ved spørsmål som var spisset til spesifikke userstories, ble de naturligvis endret mellom hver brukertest. Analyseringen av disse brukertestene bidro til å fikse feil i programmet, samt forbedre den generelle brukeropplevelsen.

##### Brukertest

---

Før brukertesten ble utført ble det laget en mal som ble brukt til utførelse av brukertesten. Dette slik at brukertest, 1 og 2 skulle være konsekvent i dens utførelse. Dette skulle resultere i en tydelig måling av forbedring mellom brukertestene. Dette er spørsmålene som ble stilt:

- Hvordan er førsteinntrykket når man åpner siden?
- Er det noe du ser kunne vært endret?
- Hva tenker du om [Relevant funksjonalitet for sprinten]?
- Tanker om [Annen relevant funksjonalitet for sprinten]?
- Har du andre bemerkninger?

## 3.4 Scrum

### 3.4.1 Sprintrutine

Sprintplanleggingen startet med et møte med produkteierne, der prioritering av user-stories sto sentralt. Et issueboard ble satt opp for å organisere og kategorisere arbeidsoppgavene. De daglige standup-møtene holdt teamet oppdatert på hva som har blitt gjort og hva som gjenstår. Det ble også registrert burndown for å overvåke effektivitet og fremgang. Sprint-review ble gjennomført sammen med produkteierne, i tillegg til at det ble utført tilstandssjekker med alle teammedlemmene for å evaluere den aktuelle situasjonen. Komfortnivået ble målt anonymt for å sikre gyldighet. Sprinten avsluttes med et retrospektiv hvor teammedlemmene diskuterte forbedringspotensialer og den kunnskapen som ble opparbeidet, noe som tas med videre til planlegging av neste sprint.

### 3.4.2 Sprintplanlegging

#### Sprint 1

Etter møtet med produkteiere begynte prosessen med å sette opp Sprint 1 issueboard. Sprint 1 fikk gul lapp, med rød tusj for backend og grønn tusj for frontend. For å forsøke å få god tidsberegnning på artifaktene ble det utført tidspoker, der hvert medlem sendte inn en beregning av hvor lang tid de ulike issuene ville ta. Dette ble utført et fåtall ganger på et bredt spekter av issues, slik at det kunne bli brukt som en basis for videre tidsberegnning på issuene, uten at alle teammedlemmene måtte stemme på hver eneste issue. Videre ble disse issuene plassert på det fysiske issueboardet.

#### Sprint 2

Sprint 2 planleggingen baserte seg på både tilbakemeldinger fra produkteiere etter sprint 1 review, samt tilbakemeldingene fra sprint 1 retrospektiv. Etter å ha fullført et møte med produkteiere, ble de tilsendt en skriftlig forklaring av hva plan var. Denne bekriftet de, før teamet deretter begynte med å sette opp issues for sprint 2:

Basert på vår erfaring fra sprint 1, brukte teamet enda mer tid på å utarbeide hvilke issues som var nødvendige for å ferdigstille produktet.

De prioriterte userstoryene, basert på produkteiers prioriteringer, kan sees i figur 4. Disse userstoriene er naturligvis i tillegg til de fra sprint 1. Issuene teamet satte opp dekket de ulike målene som hadde blitt satt basert på de prioriterte userstoriene.



Figur 1: Sprint 2 issueboard, dag 1

### 3.4.3 Standup

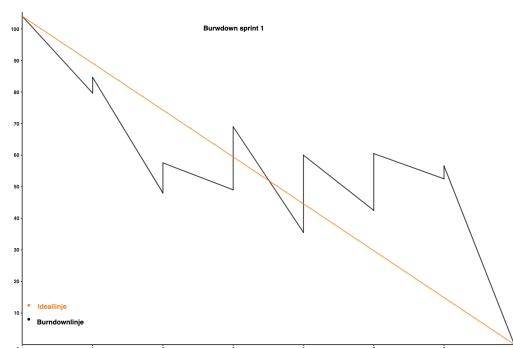
**Sprint 1:** Teamet utførte daglig standup hvor hver deltaker presenterte hva den hadde arbeidet med, etterfulgt av å presentere hva den skulle arbeide med for dagen. I tilfeller der noen hadde fullført noe dagen før, ble denne prosessen også brukt for å kartlegge hva det neste issuet burde være for teammedlemmet. Ved enkelte tilfeller ble standup også en mulighet for å assistere teammedlemmer som slet med ulike problemer, eller sette team-medlemmene sammen i grupper slik at de kunne samarbeide på ulike saker.

**Sprint 2:** Standups ble holdt på samme vis som i sprint 1, med et fokus på å orientere alle medlemmene om hva som er fokuset for dagen. Et helhetlig mål ble presentert, slik at alle visste hva dagens agenda var. Enkelte tilfeller ble det utført flere standuper gjennom dagen for å få bedre kontroll på hva som ble arbeidet på.

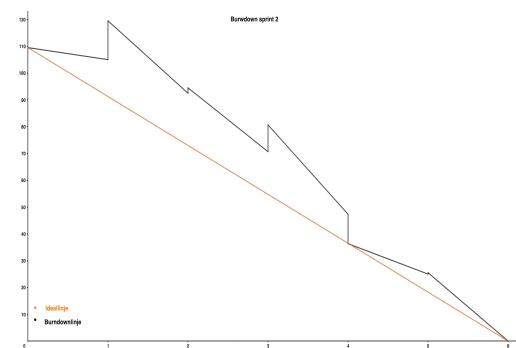
### 3.4.4 Burndown

**Sprint 1:** Gjennom hver dag av prosjektet ble det utført en burndown der teamet telte hvor mye arbeid som var utført, og hvor mye tid som var resterende. Dette bidro til at å kunne vurdere arbeidskapasiteten på teamet, og alltid vite hvor mye arbeid som var resterende.

**Sprint 2:** Dette er burndown fra sprint 2. Denne ble forbedret kraftig fra sprint 1, da teamet ble sterkere på både tidsberegning og å lage beskrivende issues slik at det ikke ble nødvendig å legge til mange flere issues etterhvert under arbeidet.



Figur 2: Burndown sprint 1



Figur 3: Burndown sprint 2

### 3.4.5 Sprint retrospektiv

Sprint retrospektiv ble startet med å utføre en undersøkelse der det ble spurta om teamet ville gjennomføre sprinten på samme måte. Dette ble fulgt opp med å utføre en digital innsending hvor teammedlemmene skulle besvare ulike spørsmål. Disse spørsmålene var:

- Hva gikk bra?
- Hva kan forbedres?
- Hva oppsummerer Sprint X?

Etter å sendt inn de ulike svarene, fikk hvert teammedlem muligheten til å fordele 2 poeng hver til de forslagene som de mener er de viktigste. Teamet valgte å utføre retrospektiv på denne måten grunnet den digitale deltakeren. Teamet valgte å implementere poengsystemet da det ga muligheten for teamet å reflektere rundt hva de aller helst ønsker å prioritere. De prioriterte tilbakemeldingene ble tatt i betraktning og tatt med videre i planlegging av neste sprint. Denne utførelsen ble gjentatt på begge sprintene. Forbedringspunktene ble deretter skrevet ned på fysiske lapper og plassert på veggen for å visualisere retrospektivet på samme måte som det fysiske issueboardet.

---

## 3.5 Rammeverk, APIer og biblioteker

### 3.5.1 Klientside

På klientsiden ble brukergrensesnittet til applikasjonen utviklet ved hjelp av Vue 3, et JavaScript-rammeverk som er ideelt for webutvikling. Det ble benyttet NPM for effektiv håndtering av prosjekts avhengigheter. I tillegg ble det brukt Vite, et moderne byggeverktøy som tilbyr rask omlasting av modular og optimalisert bygging for produksjon. Deretter ble det integrert OpenAI's API for å tilby generering av tilpassede utfordringer basert på brukerkonfigurasjoner.

### 3.5.2 Tjenereside

Tjeneresiden av prosjektet ble utviklet i Java 17 og benytter Spring Boot versjon 3. Dette rammeverket støtter utvikling av REST APIer og håndtering av serverlogikk. For å administrere databasetilgang og håndtere relasjoner mellom databasetabeller, anvendes det JPA sammen med Hibernate. Dette gir en robust løsning for ORM som forenkler datamodellering og transaksjonsstyring.

Det integreres også en MySQL-database for sikker og effektiv lagring av data. For å beskytte applikasjonen mot vanlige sikkerhetstrusler følges sikkerhetspraksiser anbefalt av OWASP, noe som inkluderer implementering av sikkerhetsmekanismer gjennom Spring Security. Dette rammeverket tilbyr støtte for autentisering og autorisasjon, sammen med beskyttelse mot vanlige angrep som SQL-injeksjon og CSRF.

Videre benyttes JWT for å håndtere brukersesjoner på en sikker måte, som bidrar til å sikre at hver sesjon er verifisert. CORS er konfigurert nøyne for å tillate forespørsler fra godkjente domener, sikre interaksjoner med APIet fra ulike klienter.

Det ble også implementert to-faktor autentisering (2FA) ved bruk av e-postbekreftelser for registrering og glemt passord. Dette sikrer et ekstra lag av sikkerhet ved å kreve bekreftelse fra brukerens e-post for å fullføre sensitiv funksjonalitet.

## 3.6 CI/CD

I byggestadiet kompileres backend med Maven og frontend bygges med Node.js. For testing er det separate jobber som kjører enhetstester på frontend ved hjelp av Vitest, samt E2E tester med Cypress. Backend-testene kjøres også i dette stadiet med Maven.

Ved fullført testing, pakkes backenden til en kjørbar JAR-fil i pakkestadiet. Hver av disse stadiene kjøres basert på regler definert for utviklings- (alle branchene) og hovedbranchene ('main' og 'dev').

Til slutt implementeres applikasjonen til produksjonsmiljøet, men dette trinnet kjøres kun når endringene skjer i "main". Denne automatiserte pipelinens sikrer en smidig og kontinuerlig integrasjon og levering av prosjektet.

## 3.7 Dokumentasjon

Backend er dokumentert med Javadoc, og APIet med Swagger. Klientsiden er dokumentert med JSDoc.

## 3.8 Byggeverktøy

På tjeneresiden ble Maven benyttet som det primære byggeverktøyet. Maven bidro til en mer effektiv arbeidsflyt ved å integrere med kontinuerlig integrasjon (CI) verktøy, noe som støttet vår praksis med kontinuerlig utvikling og testing.

---

### 3.9 Utviklingsverktøy

Under utviklingen av prosjektet ble det brukt en rekke verktøy for å forenkle prosessen på både klient- og tjeneresiden. På klientsiden ble Vite benyttet for å etablere en lokal utviklingsserver og håndtere nedlasting av nødvendige avhengigheter og pakker.

På tjeneresiden ble det brukt Make sammen med Docker Compose og Docker for å drive en lokal MySQL-database. Denne løsningen gjorde det mulig for utviklerne å enkelt sette opp og vedlikeholde databasen.

### 3.10 Versjonskontroll

Gjennom prosjektet ble Git benyttet som versjonskontrollsyste, administrert via IDI sin GitLab. Det ble valgt å organisere prosjektet som et monorepo, hvor all kode for både klient- og tjeneresiden samt konfigurasjoner og dokumentasjon befant seg i samme kodebase. Dette forenklet prosessene rundt branching, pull requests og håndtering av avhengigheter, da alle endringer og oppdateringer var sentralisert på ett sted. Dette ble godkjent av faglærer tidlig i prosjektet, og ble beskrevet som god practice.

Det ble benyttet issue-branching hvor hvert issue ble utviklet på en egen branch, noe som minimerer merge-konflikter og holder hovedbranchen stabil. Ved å bruke *conventional commits*, ble det sørget for ryddige og tydelige commitmeldinger, med bruk av både tags og gitmojis for å effektivt kunne lese og identifisere commit-logger.

### 3.11 Arbeidsfordeling

Elias fungerte som Scrum Master og sørget for at Scrum-metodikken ble fulgt. I tillegg ble det delegert forhåndsdefinerte roller som backend-ansvarlig, frontend-ansvarlig og referent. Disse rollene var ikke absolutte, og teammedlemmer bidro på ulike felter gjennom hele prosjektet. Tildeling av disse rollene, med unntak av Scrum Master, var hovedsakelig for å danne et sikkerhetsnett av ansvarstakelse. Teammedlemmene bidro på de nødvendige feltene gjennom prosessen, basert på situasjonene som oppsto. Dette fleksible rollemønsteret tillot tilpasning etter behov og sikre en smidig prosjektgjennomføring.

### 3.12 Sikkerhet

Sikkerhet var viktig gjennom hele prosjektet, og teamet fokuserte OWASP listen over sikkerhetsrisikoer. Ved å bruke JPA sammen med Hibernate som ORM, ble muligheten for injeksjoner i applikasjonen eliminert. Teamet benyttet også miljøvariabler for å unngå å lagre sensitiv informasjon direkte i kildekoden (T. O. Foundation 2024).

I tillegg til grunnleggende sikkerhetstiltak, inneholdt autentiseringstjenesten flere nøkkelkomponenter for å sikre brukeridentitet og begrense tilgangen til autoriserte brukere. Ved registrering kreves to-faktor autentisering gjennom e-post, noe som ga et ekstra lag av bruker-autentisering.

Etter vellykket autentisering, genereres en JWT som sendes tilbake til klienten som en sikker cookie. Denne tokenen er viktig for å opprettholde en sikker sesjon og sikrer at alle etterfølgende forespørsler er autorisert og autentisert.

CORS ble konfigurert for å tillate forespørsler fra godkjente kilder, noe som er essensielt for å håndtere API-forespørsler fra ulike klienter sikkerhetsmessig.

---

## 3.13 Modellering og design

### 3.13.1 Prototyper

For å teste oppsett, navigasjon og interaksjon rundt programmet ble det utviklet et wireframe. Dette wireframet ble utviklet i figma. Teamet utnyttet frames, og koblet opp de ulike delene slik at programmet kunne navigeres. Dette skapte et felles blikk på hvordan applikasjonen skulle se ut og navigeres, noe som økte effektivitet og minimerte forvirring. Se Figur 5 i Vedlegg 14 for å se noen av de mest prominente sidene og komponentene i applikasjonen fra wireframen.

### 3.13.2 API Design

På tjenersiden ble det benyttet Spring Boot versjon 3 for å utvikle API-et, som var basert på RESTful prinsipper. Dette valget muliggjorde effektiv håndtering av HTTP-forespørsler og responsflyt, og sikret at API-et var godt strukturert og enkelt å integrere med webklienten. Datautvekslingen ble utført i JSON-format, som er lett å lese og parse både for mennesker og maskiner, noe som forbedrer brukervennligheten.

## 3.14 Testing

### 3.14.1 Klientside

Klientsiden ble testet av med bruk av både Vitester for enhetstester, og cypress for E2E-tester. Enhetstestene forsikret stabilitet og korrekt informasjonsformidling i komponentene. E2E-testene forsikret at felter som logg inn og lag bruker-feltene, samt annen brukerinteraksjon, fungerte slik de skulle.

### 3.14.2 Tjenersiden

For å sikre funksjonalitet og robusthet i backend, ble det satt opp automatiserte tester ved bruk av JUnit og Mockito. JUnit ble brukt for å strukturere testtilfellene og for å utføre enhetstester av individuelle komponenter uten avhengighet til eksterne elementer. Mockito ble brukt for å simulere avhengigheter slik at teamet kunne isolere og teste spesifikke funksjoner i applikasjonen. Dette skapte kontroll over testmiljøet og mer pålitelige tester.

---

## 4 Resultater

### 4.1 Bemerkning

Grunnet problemer med kompilering av Docker Compose på enheter som ikke har .env filen i Frontend-mappen er det nødt til å legge til denne linjen i Frontend sin Dockerfile for å kunne bruke KI-generering av utfordringer. KI-generering vil fungere helt fint uten dette hvis man kjører frontenden i et lokalt miljø via npm run dev. Linjen skal inkluderes under alle de andre COPY linjene, rett før RUN npm run build”

```
COPY .env ./
```

### 4.2 Prosjektresultater

#### 4.2.1 Akseptansekriterier og oppnåelse

På de ulike userstoriene var det visse akseptansekriterier som ble satt gjennom den muntlige diskusjonen med produkteierne. Disse akseptansekriteriene presiserte kvaliteten og detaljene på de ulike ønskede funksjonalitetene:

Tabell 1: User story oppnåelser (Sprint 1)

User story (prioritering)	Akseptansekriterier	Oppnåelse
Create goals (høy)	Kunne skape mål med utfordringer, bilde, tittel, beskrivelse og sparemål	Full oppnåelse
Custom challenges (høy)	Kunne lage egne utfordringer, velge populære utfordringer eller KI-genererer utfordringer	Full oppnåelse
Visualization (mobile) (høy)	Være funksjonell på mobil, samt at progressjon visualiseres	Full oppnåelse
Social page (medium)	Kunne legge ut innlegg og se venners innlegg. Automatisk legge ut et fullført mål	Full oppnåelse

#### 4.2.2 Tilbakemelding fra produkteiere (Sprint 1):

Veldig fornøyd med skapelsen av mål. Muligheten til å lage egne utfordringer er bra. Mobilvisningen er bra på sidene det er utført på, men det kreves mer skalering av visse elementer. De er enige i prinsippet bak hvordan den sosiale siden skal fungere, men det er ingen reell funksjonalitet her enda. De liker den visuelle identiteten, og er veldig fornøyd med logoen, men ønsker at nettsiden skal være på norsk. De er fornøyd med stien som viser fremgang, men ønsker at den skal være mer tydelig. De ønsker seg mer gamification slik at det blir mer gøy å bruke nettsiden.

---

Tabell 2: User Story oppnåelser (Sprint 2)

User story (prioritering)	Akseptansekriterier	Oppnåelse
Shared goals (høy)	Kunne dele et mål med en venn. Kunne ha hver sine utfordringer i et delt mål	Full oppnåelse
Badges and streak (gamification) (høy)	Kunne oppnå medaljer basert på totale penger spart, og få en streak ved daglig oppnåelse av utfordringer	Full oppnåelse
Share challenge (medium)	Kunne dele en utfordring.	Delvis oppnåelse. En bruker kan poste et innlegg om hvilken utfordring den gjør, uten automatisk oppfordring.
Change habit (høy)	Kunne oppdatere sin sparevane, dedikasjon eller forbruksområder	Full oppnåelse
Simulate account (lav)	Simulere muligheten for å koble opp mot en bankkonto	Lav oppnåelse. Bruker kan se totale penger spart, som tilsvarer hva en ville ha tilgjengelig på konto.

#### 4.2.3 Tilbakemelding fra produkteiere (Sprint 2):

De er veldig fornøyd med muligheten til å dele et mål med en venn. De synes medaljer og streak øker gamification kraftig, slik at det blir svært gøy å bruke nettsiden. De synes muligheten til å legge ut et innlegg hvor en forteller om en utfordring en tar er en god nok løsning på kravet om å kunne dele en utfordring. Muligheten til å kunne oppdatere vanene sine er veldig intuitiv. Siden spørreundersøkelsen er så kort, er det like greit å gjøre hele undersøkelsen på nytt, da det er mulig det vekker tanker om å oppdatere noe annet samtidig. De er fornøyd med at man kan se totale penger spart, og bekrefter at simulering av konto skulle nedprioriteres i bytte med et fokus på det sosiale aspektet.

#### 4.2.4 Produkteinieres prioriteringer:

#### 4.2.5 Brukertester

Brukertestene utført med veileder ga teamet nyttig innsikt i forbedringer til programmet. Veilederen utførte brukertesten både i form av å simulere en gjennomsnittlig bruker”, samt som en med kompetanse i utvikling. Både veilederens erfaring, samt fordelen av å få ferske øyne på programmet, ga vinklinger som teammedlemmer selv ville oversett. En detaljert beskrivelse av tilbakemeldingene fra veilederen kan ses i vedleggene 16.

#### 4.2.6 Systemarkitektur

Applikasjonens arkitektur er lagdelt, designet for å støtte en dynamisk og responsiv brukeropplevelse. Ved brukerinteraksjoner hentes og oppdateres data dynamisk uten behov for å laste inn siden på nytt.

På serversiden håndterer kontrollerne forespørsler, fungere som et bindeledd mellom klienten og forretningslogikklaget. Disse kontrollerne tar imot HTTP-forespørsler, anvender forretningslogikk gjennom services, og returnerer responser i form av dataoverføringsobjekter (DTOs). For å sikre en effektiv dataflyt og beskyttelse av datalaget, bruker kontrollerne objektmappere (mappers) for å transformere data mellom databaseentiteter og DTOs. Dette sørger for at brukerne kun får tilgang til nødvendig informasjon, og at sensitive data håndteres forsvarlig.

Repositorylaget spiller en kritisk rolle ved å administrere all databaseinteraksjon. Det håndterer alle aspekter av datapersistering og transaksjonsstyring, noe som understøtter systemets dataintegritet og ytelse. Repositorylaget sikrer en skalerbar og vedlikeholdbar kodebase, som lett kan tilpasses etter hvert som applikasjonens behov utvikler seg.

Ved å benytte denne lagdelte arkitekturen kan applikasjonen enkelt skaleres og vedlikeholdes, samtidig som den opprettholder høy ytelse og sikkerhet.

#### 4.2.7 Testing

Dekningsgraden av tester på både klient- og tjenersiden tilstrekker de tekniske kravene gitt i oppgaven.

#### Klient

Klientsiden av applikasjonen oppnådde testing av 36,17% av funksjonene og 82,13% av statements. Det ble testet service, store, komponenter og views. Det ble også utført E2E Cypress tester på lag bruker, logg inn og to-faktor bekrefteelse.

#### Tjener

På tjenersiden var det et krav om 50% dekningsgrad, noe som ble oppnådd ved hjelp av enhets-tester med Mockito. I Figur ?? vises antallet backend-tester som ble utført, mens Figur ?? viser dekningsgraden på disse testene.

#### 4.2.8 Samarbeid med tredjeparts API

##### Azure Blob

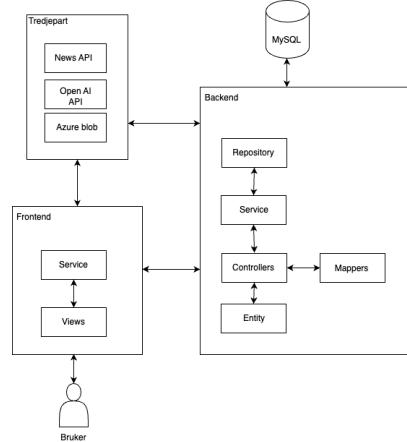
Det ble brukt Azure blob for å lagre bildene som en bruker lastet opp, enten i form av profilbilde eller bilde på et mål. Denne implementasjonen forsikrer sikker bevarelse av bildene.

##### OpenAI

Gjennom OpenAI sin GPT-3.5 API, ble det generert forslag til spareutfordringer til brukeren. Disse svarene baserte seg på dedikasjons-, erfarings- og vaneundersøkelsen som alle brukerene av programmet gjennomgår ved brukerskapelse. Dette forsikret at spareutfordringene var spisset til brukeren, og dermed svært relevante.

##### NewsAPI.org

For å gi brukeren av programmet nyttig innsikt i det økonomiske nyhetsbildet, ble det utnyttet en NewsAPI som ble spisset til å vise økonomirelaterte nyheter. Dette ble presentert på nyhetssiden i applikasjonen.



Figur 4: Systemarkitektur

---

## 4.3 Administrative resultater

### 4.3.1 Digital deltaker

Teamet hadde en spesiell situasjon, da teamet opererte med en digital deltaker gjennom 2,5 av de 3 ukene prosjektet varte. Dette påvirket deler av scrum-prosessen, og resulterte i at retrospektivseremoniene ble utført ved bruk av digitale verktøy. Dette hindret ikke selve prosessen for å innhente tilbakemeldinger og utarbeide forbedringspotensialer. Ulempene som ble oppdaget, var at de prosessene som skulle være fysiske, måtte enten gjøres dobbel eller kun utføres digitalt. Et eksempel på dette er at det fysiske issue-boardet kan ha en positiv innvirkning på gruppens motivasjon, noe dette medlemmet mistet mye av. Ved digitale retrospektiver måtte resultatene skrives på lapper i etterkant, slik at de også kunne bli vist på veggen for fremtidig forbedring.

### 4.3.2 Teamutvikling

Gjennom hele prosessen har teamet hatt en høy arbeidsmoral og et mål om å skape et flott resultat. For å oppnå dette arbeidet teamet i lange perioder. På sprint-retrospektiv ble det gitt tilbakemeldinger internt om en oppfordring til å ta flere pauser. Grunnet den korte tidsperioden prosjektet befant seg i, var det ulik interesse for disse pausene. For å opprettholde god team-moral, kunne de ulike deltakerne fritt velge når de tok pauser, uten at det fikk negative tilbakemeldinger fra andre deltakere. Denne balansen resulterte i at medlemmene fikk utført arbeidsmengden de selv ønsket. Teamet bygde samhold gjennom prosessen ved å både dra på butikken sammen i lunsjtidene, bli bedre kjent med hverandre, samt bestille pizza sammen. Disse aktivitetene gjorde teamet tettere, og bidro til et bra arbeidsmiljø.

### 4.3.3 Utførte sprintretrospektiver

Sprint retrospektivene ga nyttig innsikt i styrker og forbedringspotensialer. Utnyttelsen av poengfordeling på de prioriterte tilbakemeldingene skapte en mye mer spisset og ryddig forståelse av hva som måtte forbedres. Tilbakemeldinger fra sprint 1 ble tatt i betrakning, spesielt det å skape mer detaljerte issues med mer generøs tidsberegning. Dette resulterte i at sprint 2 retrospektiven for det meste omhandlet arbeidsbalanse, samt noen kommunikasjonsrelaterte tilbakemeldinger. Det helhetlige inntrykket var at retrospektiven bidro til å forbedre prosessen betraktelig.

For en mer detaljert beskrivelse av retrospektivene kan en se i vedleggene. Her kan en se de prioriterte situasjonene, samt ”hva som oppsummerte sprint X”.

### 4.3.4 Sprintkøen

Prosjektets backlog deles inn i sprint 1 og sprint 2. Under skildres den ferdige sprintkøen etter de to sprintene. De viktigste er beskrevet.

#### Sprint 1:

- *Implement Grafana*
- *See user statistics*
- *Implement challenge templates*
- ***Handle duplicate challenges:*** omhandler muligheten til å gjennkjenne om en utfordring ligner på en allerede eksisterende utfordring. Et eksempel på dette er ”Slutte å snuse” og ”Ikke snuse”, kan sees på som samme utfordring, og dermed var målet å kunne gjennkjenne og fjerne dette.

---

## Sprint 2:

- **Share challenge on creation:** beskriver en situasjon der en bruker lager en personlig utfordring, og ønsker å dele den med en venn. I implementasjonen har en bruker muligheten til å dele et innlegg med valgfritt innhold. En fremtidig utvikling ville automatisert deler av prosessen.
- **Show previous challenges:** beskriver en funksjonalitet som tillater en bruker å gjennomgå tidligere dagers utfordringer, i fall de for eksempel glemte å registrere en oppnåelse i løpet av dagen. Dette er det tilgjengelig backend funksjonalitet for, men på grunn av mulighetene for både å jukse og å kunne utføre kraftige modifikasjoner ved uhell, valgte teamet å utsette implementasjonen.
- **Add admin panel:** omhandler hvordan backend har full funksjonalitet for admintilgang, men ikke et utviklet panel i frontend. Dette panelet ville gitt muligheten for å modifisere og moderere populære utfordringer, samt slette personlige utfordringer og slette eller moderere brukere.

### Underveisvurdering

*Newsfeed* lå lenge i backloggen for sprint 1, men teamet oppdaget i sprint 2 at det var et hurtigere problem å løse enn først forutsett. Dermed ble det plassert *in progress* og fullført uten å negativt påvirke de andre oppgavene som måtte utføres. Slike vurderinger ble kontinuerlig utført for å maksimere kvaliteten på applikasjonen uten å svekke alleredebestemte issues.

#### 4.3.5 Burndown-diagram

Ideallinjen representerer den optimale fremgangen for å fullføre sprintens oppgaver innen sprintens slutt. Som en kan se i figur 2 viser diagrammet at det var perioder hvor teamets fremgang ikke var tett opp mot ideell fremgang. Dette skyldes både feilberegning av tid til gjennomførelse av oppgaver, samt svak planlegging før sprintens start. Noe som resulterte i at det ble lagt til flere oppgaver underveis i sprinten. Dette kan en se i figur 2 hvor burndown-linjen får en oppgang hvor det har blitt lagt til nye oppgaver.

I figur 3 kan en se en mindre svingninger i burndown-linjer, noe som gjør at den følger ideal-linjen godt. Dette kommer av bedre planlegging samt bedre analyse av tidsforbruk av oppgavene før sprinten.

#### 4.3.6 Bussfaktor

Bussfaktor referer til risikoen rundt sårbarheten av prosjektet basert på ekspertisekomtesanse som kun er tilgjengelig fra noen av deltakerne i prosjektet.

Grunnet ulike kompetanse blant deltakerne i gruppen, var det ulike deler av arbeidet som ikke kunne bli utført av alle medlemmene. Dette ledet til at spesielt integrasjonsfasen mellom front- og backend ble større samarbeidsprosjekter der deltakere lærte av andre deltakere med mer erfaring.

Ved andre tilfeller, i situasjoner hvor tiden var knapp, ble de med størst kompetanse satt til å ferdigstille viktige implementasjoner, mens andre deltakere prioriterte kommentering og dokumentering av kode, samt systemutviklingskriving og annen dokumentasjon.

---

## 5 Konklusjon og videre arbeid

### 5.1 Videre arbeid

#### 5.1.1 Ferdigstille tekniske krav

Produktet som ble levert har ingen åpenbare feil eller uferdig implementasjon. Det er flere tilgjengelige videreutviklinger i backend som det kan utvikles videre funksjonalitet for. Dette er ikke grunnet uferdig kode, men at teamet valgte å bygge opp systemet fra grunnen av med mest mulig fokus på skalerbarhet og videre utvikling. På grunn av dette er det blant annet backend-funksjonalitet for både en admin-side, samt muligheten til å reagere på innlegg. Implementasjoner med lav prioritet har ferdig funksjonalitet (for eksempel vise totale penger spart), men det er enkelt mulig å videreføre dette til en større implementasjon om ønsket.

#### 5.1.2 GDPR og personvern

Grunnet vår plan om at applikasjonen vår ikke skulle ut i produksjon, er det visse funksjonaliteter som mangler for å følge GDPRs regler rundt behandling av persondata. I frontend er det ikke muligheten til å slette en bruker, og heller ikke en mulighet til å fjerne all brukerdata. Denne funksjonaliteten er naturligvis tilgjengelig i backend, og er dermed en naturlig del av en videre utvikling som fører til en produksjonsvennlig applikasjon.

Det lagres ikke data som ikke er nødvendig for programmets funksjonalitet, og applikasjonen lagrer kun informasjon som brukeren selv bekrefter ved å trykk på *lag bruker*.

Grunnet teamets fokus på en sosial plattform, ble nedprioritering av BankID videre relevant, med tanke på GDPR. Det å skape en sosial plattform med personnummeridentifisering kan skape flere ulike sikkerhetsrisikoer som teamet valgte å eliminere ved å prioritere en solid registrering med en robust *to-faktor bekrefteelse*.

Disse API-nøklene er ikke skalerbare for en reell produksjon da kostnaden av de ulike API-nøklene kan bli for høy. Ved en reell produksjon ville teamet ønsket å inngått en avtale med de ulike selskapene som tilbyr API-ene og avtalt spesialiserte avtaler med mer gunstig skalering.

#### 5.1.3 Videre utvikling av tredjeparts API

For å skalere bruken av tredjeparts API-er, er det nødvendig å inngå et samarbeid med leverandøren av disse. Dagens løsning med én privat API-nøkkel er lite skalerbar. Ved å inngå egne samarbeidsavtaler vil man oppnå høyere kvaoter, bedre support og optimaliserte løsninger som er tilpasset behovene. Disse tiltakene vil bidra til å sikre at produktet kan skalere effektivt og opprettholde høy ytelse og tilgjengelighet i et produksjonsmiljø med større trafikk.

#### 5.1.4 Fremtidige implementasjoner

Grunnet teamets valg om å skape en sosial plattform, og derav bortprioritering av BankID, er det et kurrant felt for videre utvikling. Vår implementasjon av funksjonaliteten vår baserer seg på at en bruker bekrefter eller dokumenterer forbruk selv, med fokus på å gjøre dette så enkelt og intuitivt som mulig. En fremtidig implementasjon, med fokus på oppkobling mot en reell forbrukskonto, ville også ledet til en prioritering av BankID. Dette ville blitt en stor endring i fokuset på applikasjonen, og ble nedprioritert av produkteierne, men kan likevel vurderes for fremtidig implementering.

---

### 5.1.5 Videre prosessarbeid

Gjennom sprintene opparbeidet teamet større kunnskap rundt hvordan de arbeider optimalt. Det ble gjort forbedringer av både issues, burndown-diagrammet, kommunikasjon både internt og med kunde, og fortsatte å opprettholde et effektivt og komfortabelt arbeidsmiljø. Ved potensielle videre sprinter ville teamet forbedret seg enda mer rundt beskrivelsen og tidsberegningen av issues. Arbeidstempoet ville blitt redusert generelt, men teamet ville opprettholdt motivasjonen og engasjementet rundt å utvikle interessante funksjonaliteter.

### 5.1.6 Drøfting

Sluttproduktet er en velfungerende applikasjon med moderne implementasjoner. Produktene ble svært fornøyd med det endelige resultatet, og beskrev det som både penere og mer funksjonelt enn det de forrutså. Alle prioriterte userstories ble oppfylt, dog noen mindre prioriterte userstories ble implementert av svakere grad enn ønsket. Resultatene som ble oppnådd kom fra hardt arbeid, dedikerte teammedlemmer og tydelig kravspesifisering fra begge parter. Fokuset på et sosialt medie som tilbyr sparing ga teammedlemmene en motivasjon som skapte et flott resultat. Det er mulig at et veldig teknisk fokus ville skapt mindre engasjement blant både teammedlemmene og produktene. Den unyttede teknologien tilbydde muligheten for et stabilt, sikkert og moderne resultat som alle parter ble fornøyd med. Grunnet applikasjonens størrelse og scope er det naturligvis oppryddinger og effektiviseringer som kan forbedre både ytelse og kvalitet, men dette kan sees på som kontinuerlig forbedrende.

## 5.2 Konklusjon

Sparesti er en applikasjon som sterkt ligner på produktene sin visjon. Applikasjonen dekker alle prioriterte brukerhistorier med høy og medium prioritet, og dekker grunnleggende funksjonalitet for brukerhistoriene med lav prioritet. Teamet er stolte av funksjonaliteten som har blitt implementert på kort tid. Teamet har blitt utfordret, og forbedret seg i, kommunikasjon, tidsberegning og teknisk kodekompetanse. Prosjektet har vært svært lærerikt og engasjerende for gruppen.

Produktene opprettholdt tydelig og presis kommunikasjon gjennom hele prosessen. De var konsistente og konsekvent i deres ønsker og tilbakemeldinger, noe som skapte mye selvtillit rundt utviklingen av funksjonalitetene. De var motiverte og tilgjengelige gjennom mail, og møtte opp presist til alle avtaler. De tok også med seg godt humør inn i et rom fylt av slitne teammedlemmer.

---

## Referanser

- Alliance, Agile (2024). *User Stories*. URL: <https://www.agilealliance.org/glossary/user-stories/> (sjekket 10. mai 2024).
- Bass, J. (nov. 2022). *What is CI/CD? Continuous Integration and Continuous Delivery Explained*. URL: <https://www.redhat.com/en/topics/devops/what-is-ci-cd> (sjekket 10. mai 2024).
- Datatilsynet (2023). *Om personopplysningsloven med forordning og når den gjelder*. URL: <https://www.datatilsynet.no/regelverk-og-verktøy/lover-og-regler/om-personopplysningsloven-og-narden-gjelder/> (sjekket 9. mai 2024).
- Ellingwood, Justin (2024). *Prisma. What is an ORM?* URL: <https://www.prisma.io/dataguide/types/relational/whatis-an-orm> (sjekket 10. mai 2024).
- Foundation, Interaction Design (2024). *User Experience (UX) Design*. Accessed: May 12, 2024. URL: <https://www.interaction-design.org/literature/topics/ux-design>.
- Foundation, The OWASP (2024). *OWASP Top Ten*. Accessed: 2024-05-12. URL: <https://owasp.org/www-project-top-ten/>.
- geeks, Geeks for (mar. 2023). *How to Write Test Cases in Java Application using Mockito and Junit?* URL: <https://www.geeksforgeeks.org/how-to-write-test-cases-in-java-application-using-mockito-and-junit/> (sjekket 10. mai 2024).
- (apr. 2024). *Unit Testing - Software Testing*. URL: <https://www.geeksforgeeks.org/unit-testing-software-testing/> (sjekket 10. mai 2024).
- IBM (2024). *What is a REST API?* URL: <https://www.ibm.com/topics/rest-apis> (sjekket 10. mai 2024).
- Kong (jun. 2023). *Understand the Differences; API Authentication vs API Authorization*. URL: <https://konghq.com/blog/engineering/api-authentication-vs-api-authorization> (sjekket 10. mai 2024).
- Microsoft (2023). *End-to-End Testing*. Accessed: May 12, 2024. URL: <https://microsoft.github.io/code-with-engineering-playbook/automated-testing/e2e-testing/>.
- Oracle (nov. 2020). *What is a Database?* URL: <https://www.oracle.com/database/what-is-database/> (sjekket 10. mai 2024).
- Perforce (mar. 2023). *What Is a Monorepo?* Accessed: May 10, 2024. URL: <https://www.perforce.com/blog/vcs/what-monorepo>.
- Scrum.org (2024a). *What is a Daily Scrum?* URL: <https://www.scrum.org/resources/what-is-a-daily-scrum> (sjekket 10. mai 2024).
- (2024b). *What is a Product Backlog?* URL: <https://www.scrum.org/learning-series/what-is-scrum/the-scrum-artifacts/what-is-a-product-backlog> (sjekket 9. mai 2024).
- (2024c). *What is a Sprint backlog?* URL: <https://www.scrum.org/learning-series/what-is-scrum/the-scrum-artifacts/what-is-a-sprint-backlog> (sjekket 10. mai 2024).

# Vedlegg

1	Visjonsdokument	A1
2	GitLab lenker	B1
3	Møteinkalling fra sprint 1 planlegging	C1
4	Møtereferat fra sprint 1 planlegging	D1
5	Møteinkalling for møte med produkteier 19.04.2024	E1
6	Møtereferat for møte med produkteier 19.04.2024	F1
7	Møteinkalling for sprint 1 review 25.04.2024	G1
8	Møtereferat for review 1 25.04.2024	H1
9	Møteinkalling for sprint 2 planlegging 25.04.2024	I1
10	Møtereferat for sprint 2 planlegging 25.04.2024	J1
11	Møteinkalling for sprint 2 review 03.05.2024	K1
12	Møtereferat for sprint 2 review 03.05.2024	L1
13	Produkteiers prioriteringer	M1
14	Wireframe	N1
15	Visuell profil	O1
16	Usertest 1	P1
17	Usertest 2	Q1
18	Produktkø	R1
19	Sprintkø Sprint 1	S1
20	Sprintkø Sprint 2	T1
21	ER diagram fra 25.04	U1
22	Domenemodell	V1
23	Wiki	W1

---

## Visjonsdokument

# Visjonsdokument

IDATT2106 – Systemutvikling med smidig prosjekt

Bachelor Ingeniørfag, data

Institutt for datateknologi og informatikk (IDI)

Fakultet for informasjonsteknologi og elektronikk (IE) Norges teknisk-naturvitenskapelige universitet (NTNU)



*SpareSti er laget for å gjøre sparing til en lek. Appen er integrert med nettbanken din og har derfor oversikt over hva pengene dine brukes til og kan basert på dette gi deg persontilpassede sparetips. Appen passer for alle sparemål og gir motivasjon og tips basert på dine ønsker.*

*Siden vi vet at det kan være vanskelig å spare penger, setter SpareSti pengene automatisk inn på sparekonto når du fullfører utfordringene. Basert på oppsparte midler vil feeden gi deg personlige tips om hvordan pengene kan investeres og du vil kunne sette opp budsjett som gir deg oversikten du trenger for å gjøre bevisste valg.*

## Revisjonshistorikk

Dato	Versjon	Beskrivelse	Forfatter(e)
09.04.2024	0.1	Utfylling av produkt/visjon	Produkteiere
11.04.2024	0.2	Utbroddering av 3.3 og 5	produkteinere

---

# Innholdsfortegnelse

<b>Revisjonshistorikk</b>	<b>1</b>
<b>Innholdsfortegnelse</b>	<b>2</b>
<b>1 Innledning</b>	<b>3</b>
<b>2 Sammendrag – problem og produkt</b>	<b>3</b>
2.1 Problemsammendrag	3
2.2 Produktsammendrag	6
<b>3 Overordnet beskrivelse av interesser og brukere</b>	<b>7</b>
3.1 Oppsummering interesser	7
3.2 Brukermiljø	8
3.3 Sammendrag av brukernes behov	8
<b>4 Produktoversikt</b>	<b>10</b>
<b>5 Produktets funksjonelle egenskaper</b>	<b>10</b>
5.1 Alternative utvidelser	12
5.2 Ikke-funksjonelle egenskaper og krav	12
<b>6 Innlevering</b>	<b>13</b>
<b>7 Vedlegg og sluttrapport</b>	<b>14</b>
7.1 WIKI-struktur og innhold	15

# 1 Innledning

Dette dokumentet beskriver overordnede krav til prosjektoppgaven i emnet IDATT2106 Systemutvikling 2 med smidig prosjekt. Oppgaven består i å utvikle webapplikasjonen *SpareSti*. Prosjektet skal utføres av 2. år Bachelor ingeniørfag, data, i en intensiv periode av vårsemesteret 2024, hvor produkteierne er 1. årige masterstudenter i Digital Transformasjon.

Studentene skal utvikle en webapplikasjon hvor målet er å *motivere voksne til å forbedre økonomien sin*. Produkteierne har gjennomført en Design Thinking prosess, hvor resultatet ble en visjon for hvordan denne webapplikasjonen skal bidra til å motivere til sparing og få oversikt over eget forbruk gjennom et budsjett. I visjonsdokumentet fremkommer et problemsammendrag og krav til løsningen, samt relevant informasjon som kan bistå studentene med å fullføre produktutviklingen.

## 2 Sammendrag – problem og produkt

I dette kapitlet gis det en overordnet beskrivelse av problemet som skal løses og produktet som vil bli benyttet som løsning på dette problemet.

### 2.1 Problemsammendrag

Hva er problemet?	<p><b>Sparing:</b> Mange opplever utfordringer med å spare fordi de ikke har tilstrekkelig oversikt over egen økonomi. Dette inkluderer ofte en mangel på muligheter til å filtrere og kategorisere utgifter, noe som kan gjøre det vanskelig å identifisere mulige besparelser og optimalisere sparingen.</p> <p><b>For lite personalisert:</b> Tradisjonelle sparemetoder og verktøy tilbyr ofte generiske råd som ikke tar hensyn til den enkeltes unike finansielle situasjon og sparemål. Dette kan føre til at brukerne ikke føler at rådene er relevante for dem, noe som reduserer effektiviteten av rådene og sparingen.</p> <p><b>Motivasjon:</b> Mange mangler motivasjon til å fortsette å spare, spesielt når de ikke ser umiddelbare fordeler eller når prosessen føles monoton og uinspirerende. Dette er en barriere som kan hindre effektiv oppbygging av sparekapital.</p> <p><b>Manglende kunnskap/kompetanse:</b> En betydelig hindring for effektiv sparing og investering er mangel på nødvendig kunnskap og kompetanse. Uten forståelse om hvordan man best styrer pengene sine eller investerer dem, risikerer mange å ta dårlige finansielle beslutninger som kan påvirke deres økonomiske fremtid negativt.</p>
Hvem berøres av dette problemet?	Voksne mennesker mellom 18-60 år som ikke har kunnskap, initiativ, og/eller tid til å unngå problemene. Mangel på

	<p>effektiv sparing og dårlig finansiell oversikt er noe man kan møte på i enhver finansiell situasjon. Både i hverdagen når man handler dagligvarer, og større kostnader/investeringer over lengre tid innenfor bolig eller transport.</p> <p>Dette gjelder derfor alle voksne mennesker uansett hvilken demografi de faller innenfor når det gjelder økonomisk situasjon, utdanning, yrke og/eller nivå av kunnskap.</p>
Hva er dagens konsekvenser av dette problemet?	Disse problemene kan føre til flere konsekvenser. Blant annet kan mangel på kunnskap vedrørende personlig økonomi/sparing føre til en dårligere oversikt og kontroll. Videre kan dette resultere i dårligere økonomiske prioriteringer og dermed økt sannsynlighet for å pådra seg høy gjeld/inkasso. Disse problemene kan i tillegg gjøre det vanskelig å oppnå ønskede sparemål, på bakgrunn av manglende kunnskaper, motivasjon og liknende.
Hva kan en vellykket løsning på dette problemet bidra til?	En vellykket løsning kan bidra til å øke motivasjon knyttet til sparing, som gjør det lettere å oppnå sparemål man har satt seg. Videre vil en også tilegne seg mer kunnskap rundt eget forbruk og mulighet til å danne en bedre struktur på personlig økonomi. Det vil også skape mer bevissthet rundt egen økonomi og dermed bidra til mindre sløsing av penger.
Forretningsmuligheter?	<p>Inngå samarbeid med ulike aktører. Dette kan være banker, aviser, næringslivsaviser (E24 og DN) og andre aktuelle aktører.</p> <p>Reklame i app som er sparerelevant (f.eks. matkasse, billigere abonnement, mm.).</p> <p>Betale for reklamefri, eller premiumversjon med mer funksjonalitet. Evt. bedre oppfølging via appen, eller mulighet for flere sparemål samtidig.</p> <p>Samle SpareSti-badges eller poeng for å få rabatt hos samarbeidspartnere. Avhengig av samarbeid med ulike aktører, f.eks. elektronikk, dagligvarer, ting (nille, biltema...).</p> <p>Verve venner og få bonuser, kan samle informasjon om brukere for å benytte denne.</p>

## 2.2 Produktsammendrag

Hvem er IT-løsningen laget for?	Løsningen er laget for voksne som ønsker å gjøre mer bevisste valg for å få en oversikt over eget forbruk og få persontilpassede sparetips for å forbedre egen økonomi.
Hvilket behov har brukerne av IT-løsningen?	Brukernes behov er å få sparetips tilpasset eget forbruk. Tilpasningen baserer seg på data fra kontoutskrift og brukerens aktivitet i applikasjonen.  Videre er det hensiktsmessig å få bedre oversikt og visualisert hvor mye en kan spare, dersom man gjør små endringer i forbruket sitt. I tillegg ønsker de å motta et personlig budsjett og mulighet til å kategorisere utgifter.
Navn/arbeidsnavn for IT-løsningen (produktet)?	"SpareSti" Applikasjonen skal være tilpasset standard PC-skjerm, men skal kunne kjøre på standard mobilskjerm som applikasjon (responsivt design).  Applikasjonen skal motivere til sparing ved å spennfisere sparingsmetoder. Eksempelvis Duolingo. Brukerne skal kunne tilpasse sparing i appen ved å spesifisere behov og mål for sparingen. Det legges opp til at brukeren lager en "personlig sparesti" for å nå sine sparemål og ha det gøy i spareprosessen.
Finnes det andre løsninger som benyttes for problemer av samme art?	Bankene har forskjellige varianter av spareapper, i tillegg har de fleste banker en mobilbank som gir en oversikt over forbruk. Et eksempel på en slik spareapp er: <ul style="list-style-type: none"> <li>• Spare fra DNB (får oversikt over egne verdier - konto, fond, aksjer og pensjon)</li> <li>• Nordea og Sbanken (i mobilbanken /app) har funksjonalitet til å rekategorisere transaksjoner</li> </ul> Andre applikasjoner som skal hjelpe med økonomi er: <ul style="list-style-type: none"> <li>• Dreams (spare til det man drømmer om, kobles sammen med banken din, koster 29 kr/mnd og da får man verktøy som f. eks. å ha ubegrenset antall drømmer, appen har et klimafokus, samarbeider med Storebrand med refinansiering og lån)</li> <li>• Buddy (man kan selv sette opp et budsjett og ha oversikt over forbruk, inntekt og sparepenger, og dele det med andre - premiumabonnement for alle funksjoner)</li> </ul> Oppbygging av ny kompetanse: <ul style="list-style-type: none"> <li>• Duolingo har funksjonalitet som visualiserer hvor langt du har kommet til målet, prestasjoner (badges), poeng som opptjenes, mulighet til å knytte med andre venner. En "spareapplikasjon" kan på lignende måte</li> </ul>

	vise hvordan brukeren ligger an på veien mot sitt personlige sparemål.
Hva kjennetegner den nye IT-løsningen og på hvilken måte skiller den seg fra eventuelle andre løsninger som benyttes for problemer av samme art?	<p>Den nye IT-løsningen skal forenkle sparing for enkeltpersoner gjennom bruk av ulike "utfordringer" som bidrar til økt sparing og en budsjett funksjonalitet. Disse utfordringene er strukturert etter en sti, hvor man må gjennomføre én utfording før man kan gå løs på den neste.</p> <p>Det som skiller SpareSti fra andre, lignende applikasjoner, er at SpareSti genererer persontilpassede spareutfordringer. Dette er noe andre spareapper ikke har lagt opp til på samme måte. Disse utfordringene henter data fra ditt forbruk gjennom integrasjon med nettbanken for å kunne personaliseres. I SpareSti kan man i tillegg sette opp sparemål basert på egne preferanser. Man kan selv bestemme lengde (f.eks. en uke eller én mnd) og vanskelighetsgrad (f.eks. lett eller ekspert) på sparemålet, samt hva sparemålet spesifikt omhandler.</p> <p>SpareSti skal ha en egen funksjonalitet for budsjettering, med mulighet for å lage egne kategorier og ønsket start/sluttidspunkt på budsjettmåneden. Denne funksjonaliteten skal kunne komme med forslag til budsjett basert på tidligere transaksjoner, men brukeren skal også kunne sette opp og tilpasse et eget budsjett dersom dette er foretrukket.</p> <p>Gjennom budsjettet og de utfordringene som er knyttet til sparingen, er målet at det skal være motiverende og gøy for brukerne å spare til egne satte mål.</p>
Målformuleringer	Det overordnede resultattmålet er at SpareSti skal bidra til at brukerne sparer mest mulig penger. Effektmål går ut på at brukerne blir mer bevisst på egen økonomi og eget forbruk. Appen vil det også være tilgjengelige sparetips samt siste nytt om det økonomiske nyhetsbildet, som vil føre til økt bevisstgjøring og at brukerne blir mer oppdatert om den økonomiske situasjonen i verden. Dette vil være ytterligere effektmål.

## 3 Overordnet beskrivelse av interesser og brukere

### 3.1 Oppsummering interesser

Navn	Utdypende beskrivelse	Rolle under utviklingen
Produkteier	Representerer kunde. Student ved studiet Master i Digital Transformasjon.	Bistår med innspill og er sentrale ved prioritering av krav/ godkjenning av inkrement
Scrum Master	En av studentene i utviklerteamet	Bistår prosjektgruppa og sørger for at prosess følges
Scrum Team	Utviklere av systemet	Organiserer seg selv og er ansvarlig for selve utviklingen av systemet
Faglærer	Emnets emneansvarlig og faglærere	Veileder teamene underveis i prosessen
Brukere	Potensiell målgruppe, voksne mellom 18 og 60 år.	Bistår underveis prosjektet med å gjennomføre brukertester os.

### 3.2 Brukermiljø

Alle brukere bruker løsningen med valgfri nettleser både på PC og mobil, derfor må plattformen fungere i de vanligste nettleserne og har som krav å være mobilvennlig.

### 3.3 Sammendrag av brukernes behov

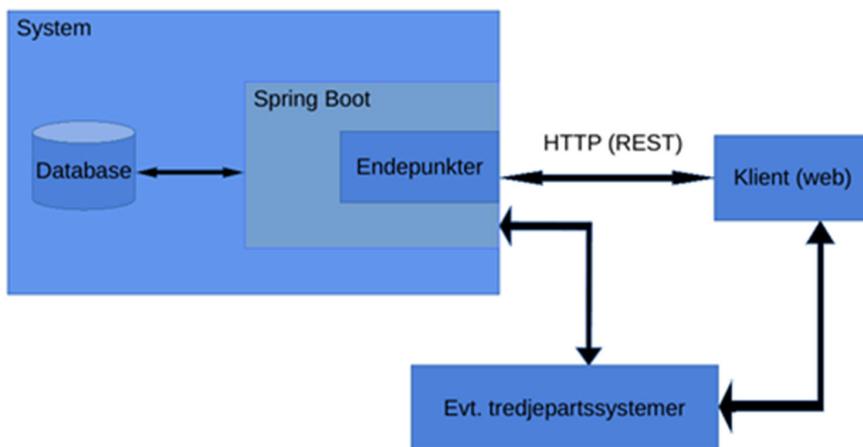
#### Ordforklaring:

- **Sparemål:** Dette er det brukeren ønsker å spare til
  - Eks: Reise til Spania
- **Spareutfordring:** Dette er det brukeren gjennomfører for å klare å spare til sparemålet.
  - Eks: Reduser kjøp av kaffe. Du pleier å kjøpe 6 kaffe på én uke. Din utfordring er å kun kjøpe 2 kaffe denne uka
- **Spareoversikt:** Gir brukeren oversikt over spareutfordringene og sparemålet

Behov	Kort beskrivelse av foreslått løsning	Prioritet av behov.
Lage sparemål	<p>Bruker skal kunne opprette et eller flere sparemål, der de setter opp</p> <ul style="list-style-type: none"> <li>• Navn på sparemål</li> <li>• Hvor mye som skal spares</li> <li>• Når målet skal være nådd</li> </ul> <p>Eksempel: Reise til Spania, 5 000 kr, 12.07</p>	Høy
Motta persontilpassede utfordringer	<p>Appen skal kunne foreslå spareutfordringer basert på brukerens forbruk, hentet fra transaksjoner i banken og hvilke utfordringer brukeren aksepterer. Brukeren velger selv</p> <ul style="list-style-type: none"> <li>• Vanskelighetsgrad</li> <li>• Lengde på utfordring (7-30 dager)</li> <li>• Å akseptere spareutfordring eller generere ny</li> </ul>	Høy
Oversikt over penger spart	<ul style="list-style-type: none"> <li>• Mulighet til å se hva brukeren har spart opp totalt ved hjelp av appen</li> <li>• Mulighet til å se hva brukeren har spart til et spesifikt mål</li> <li>• Visuell oppsummering av status på sparemål</li> </ul>	Høy
Enkelt brukergrensesnitt	Brukeren skal enkelt finne frem til funksjonalitetene den ønsker. Appen skal være oversiktlig og intuitiv.	Høy
Opprette budsjett	<ul style="list-style-type: none"> <li>• Mulighet for å sette opp budsjett med nytte kategorier</li> <li>• Se hvor mye som er igjen i budsjettet post</li> </ul>	Middels
Økt kunnskap hos brukere	Appen skal bidra til at brukerne får økt kunnskap, oversikt og kompetanse rundt personlig økonomi, derav utgifter, inntekter, aksjer og fond, o.l.	Middels
Visualisering av status på utfordringer	Brukere skal kunne se fremgang på utfordringene generert av appen. Fremgangen på utfordringene kontrolleres av bruker. Denne fremgangen vises på hovedsiden med oversikt over alle utfordringer, og inne på hver enkelt utfordring.	Middels
Badges for motivasjon og følelse av oppnåelse	<p>Appen skal inneholde oppnåelige generelle mål utenom utfordringene og sparemålene som skal motivere brukeren</p> <p>Eksempel: Gjøre flere utfordringer på rad - få streak.</p>	Middels
Sparing mot felles mål	Brukere kan lage et sparemål sammen med andre brukere, og man kan se hvor mye en partner/venn har spart mot samme mål. Man kan få forskjellige	Lav

	<p>utfordringer knyttet til hvordan hver bruker bruker sine penger.</p> <ul style="list-style-type: none"> <li>Eksempel: Et par sparer til en ferie til sommeren → bruker 1 får egne utfordringer, og kan se hvor mye bruker 2 har spart, og motsatt.</li> </ul>	
Dele spareutfordringer med andre brukere	<p>Brukeren kan utfordre venner i appen til å gjennomføre samme spareutfordring.</p> <p>Eksempel: Bruker 1 får spareutfordring om shoppestopp i 30 dager og inviterer bruker 2 til å delta i samme spareutfordring. Bruker 1 og 2 har ulike sparemål.</p>	Lav

## 4 Produktoversikt



## 5 Produktets funksjonelle egenskaper

Kort navn på funksjonell egenskap	Beskrivelse av funksjonell egenskap
Registrere bruker	<p>Registrere bruker med BankID for å koble registrert bruker opp imot bankkonto på en sikker måte.</p> <p>Mulighet til å velge innloggingsmetode: opprette egendefinert passord, fingeravtrykk, facelD eller BankID.</p> <p>Brukere skal velge hvilken konto pengene som spares skal trekkes fra (forbrukskonto), og hvilken konto pengene skal overføres til (sparekonto).</p>

Konfigurering	<ul style="list-style-type: none"> <li>• Mulighet for bruker å registrere hvor kjent de er med sparing</li> <li>• Mulighet for bruker å registrere hvor store vaneendringer brukeren er villig til å gjøre</li> <li>• Mulighet til å kartlegge spareutfordringer brukeren kunne se for seg å gjennomføre, gjennom eksempler på utfordringer.           <ul style="list-style-type: none"> <li>○ Eks: Shoppestopp, slutte å snuse, spise mer vegetar, la bilen stå, kjøp brukt, lag egen lunsj.</li> </ul> </li> </ul> <p>Dette gjøres ved første pålogging og ved behov i profil, slik at appen blir kjent med brukeren og kan gi personaliserte spareutfordringer.</p>
Login	Mulighet til å velge mellom å logge inn med et egendefinert passord, fingeravtrykk, BankID eller FacetID. Mulighet for funksjonalitet "glemt passord".
Knytte til bankkonto	Koble til en konto hvor sparepengene trekkes fra, og en konto hvor sparepengene settes inn på.
Feed for nyheter og sparetips	<ul style="list-style-type: none"> <li>• Gi brukeren sparetips basert på livssituasjon</li> <li>• Nyheter som er relevant for enkeltpersoners økonomi</li> </ul>
Budsjett	<ul style="list-style-type: none"> <li>• Appen genererer et budsjett basert på transaksjoner som er blitt gjort fra forbrukskonto.</li> <li>• Brukeren kan selv opprette eller gjøre endringer i et budsjett manuelt med egne kategorier.</li> <li>• Basert på transaksjoner fra forbrukskonto skal appen visualisere forbruket til brukeren i forhold til budsjettet.</li> </ul>
Autotrekk til sparekonto	<p>Pengene skal trekkes fra brukerens forbrukskonto når spareutfordringene registreres gjort av brukeren. Dette betyr at selv om brukeren ikke fullfører utfordringen i sin helhet, vil sparing i de utfordringene som er fullført bli overført til sparekonto.</p> <p>Eksempel:</p> <ul style="list-style-type: none"> <li>- <b>Utfordring:</b> Ikke kjøp kaffe denne uken. Brukeren kjøper normalt 6 kaffe.</li> <li>- <b>Resultat:</b> bruker kjøper 2 kaffe. Dette er fortsatt en forbedring og pengesummen som er spart på å unnlate å kjøpe 4 kaffe vil bli overført til sparekonto.</li> </ul>
Sparemål	Brukeren skal selv kunne opprette egne sparemål.
Generere spareutfordringer	Appen genererer spareutfordringer basert på brukerens valg av vanskelighetsgrad og lengde på spareutfordringen.
Spareoversikt "sparesti"	Oversikt over veien til et gitt sparemål, og utfordringer som er knyttet til dette målet. Oversikten er delt i flere trinn, slik at man kan ta for seg et gitt antall spareutfordringer for en periode (eks: Duolingo).
Spareutfordring	<p>Mulighet til å gå inn på ulike spareutfordringer i spareoversikten, slik at brukeren får informasjon over en konkret spareutfordring</p> <ul style="list-style-type: none"> <li>• Når det genereres en ny spareutfordring, vil denne siden gi brukeren mulighet til å akseptere eller avslå spareutfordringen.</li> <li>• Ellers:           <ul style="list-style-type: none"> <li>○ se prosjeksjon på spareutfordring</li> <li>○ manuelt registrering av sparing som tilhører en spareutfordring</li> </ul> </li> </ul>

	<p>Eksempel på utfordring: "Du pleier å kjøpe kaffe 6 ganger i uken. Utfordringen din er å kjøpe kun 2 kaffe i løpet av denne uken. Slik vil du kunne spare 200 kr".</p>
Gamification	<ul style="list-style-type: none"> <li>• Mulighet til å se oppspart pengesum spart til pågående sparemål uansett hvor brukeren navigerer i appen</li> <li>• Mulighet til å oppnå ulike badges som vises på profilen <ul style="list-style-type: none"> <li>○ Eksempel: Streak for å se antall utfordringer man har klart å gjennomføre etter hverandre</li> </ul> </li> <li>• <i>Progressbar</i> tilknyttet tildelt utfordring slik at brukeren ser hvordan de ligger an med utfordringen i sin personlige spareoversikt</li> <li>• Utfordringer blir satt opp i en sti med sparemålet i slutten, for å ha visualisering av prosesjon relatert til målet</li> </ul>
Profil	<p>Oversikt over:</p> <ul style="list-style-type: none"> <li>• Personlige opplysninger (navn, e-post, etc.)</li> <li>• Mulighet til å endre forbrukskonto, sparekonto og kontotilknytning</li> <li>• Prestasjoner som er gjort, som vises med badges (utfordringer gjennomført osv.)</li> <li>• Generelle innstillinger: min profil, lisenser, personvern osv.</li> <li>• Endre konfigurasjonsinnstillingar</li> </ul>

## 5.1 Alternative utvidelser

Funksjon/behov	Beskrivelse
Samarbeid med DN, E24, o.l.	Opparbeide et samarbeid som gjør at brukere kan få tilgang til diverse relevante nyhetsartikler fra aviser som DN og E24.
Dele profil med andre brukere	Brukeren skal ha mulighet til å dele profilen sin med andre brukere, både for å spare mot et felles mål eller konkurrere om utfordringer.
Mulighet til å ha flere sparemål samtidig	I første omgang kan man gi muligheten til brukeren å kun ha et sparemål om gangen. Men det vil være gunstig for brukeren å kunne spare til flere mål samtidig.
Avatar knyttet til personen	Brukeren skal kunne lage egen avatar som representerer brukeren.

## 5.2 Ikke-funksjonelle egenskaper og krav

- Det kreves programmatisk testing av koden. På serversiden skal det være minst 50% dekningsgrad, men fordelingen av hvilke typer tester (enhet vs. Integrasjon vs. E2E er opp til gruppa selv - men dere må vise at dere behersker flere typer tester og begrunne valg av tester). På klienten er det krav til minst 30 % dekningsgrad, fortrinnsvis enhetstester.
- Alle data som benyttes av applikasjonen skal lagres i skolens MySQL-database eller en i base som spinnes opp, initialiseres, og populeres med litt

- 
- eksempeldata når en kjører applikasjonen. Databasefunksjonaliteten skal tilgjengeliggjøres for klienten via REST-tjenester.
- En plattformuavhengig nettleser med støtte for en nyere HTML-standard må kunne brukes som klient mot applikasjonen. Minimumskrav er støtte for både Chromium (Chrome) og Firefox, så begge deler må sjekkes.
  - Løsningen skal være i samsvar med WCAG 2.2 prinsipp 1 (Begripelig) - og ha god brukskvalitet, den skal være tilpasset målgruppen, og lett og intuitiv å bruke. Dette skal dokumenteres.
  - Løsningen skal ha god sikkerhet. Minstekrav er implementering av autentisering og autorisering, samt at løsningen sjekkes opp mot OWASP ([OWASP Top Ten Web Application Security Risks](#)) A03:2021-Injection (merk at XSS også er en del av denne kategorien nå til dags).
  - All kode skal lagres på skolens Gitlab (<https://gitlab.idi.ntnu.no/>) – hvert scrum team oppretter et eget Gitlab-prosjekt på følgende format:  
**idatt2106\_2024\_teamnr** (små bokstaver, all tall mindre enn 10 skal 0-paddes, altså "06", ikke "6"). Faglærer Surya Kathayat, Grethe Sandstrak og Muhammad Ali Norozi skal gis Reporter-tilgang til gitlab-prosjektet. Inkluder **teamnr** og fullt navn på alle teammedlemmer på WIKI-landingsside.

Det forventes at dere benytter allerede kjente hjelpemidler/teknologier:

- Bruk av byggesystemer som kjører tester og gjør det enkelt å kjøre systemene.
- Serversiden skal være et REST-snitt og bruk av Spring Boot er påkrevd.
- Klienten skal være web-basert og en SPA (Single Page Application). Annet rammeverk enn Vue kan benyttes, så lenge alle i gruppa er enige om avgjørelsen.
- Continious Integration – CI, med testrapporter i Gitlab. Continious Delivery – CD er frivillig.
- Persistens i vanlig SQL-base. En kan velge å jobbe mot MySQL-base på NTNUs servere, eller ha base lokalt.

## 6 Innlevering

Innlevering skal gjøres i Inspera og er todelt. Del 1 er gruppeinnlevering og Del 2 er individuell innlevering.

### Del 1: Gruppeinnlevering i Inspera

Det skal leveres tre (3) filer, samt lenker til teamets git-repo:

1. Sluttrapport (1 pdf-fil)
2. Vedlegg (1 pdf-fil)
3. Programkode (1 zip-fil)

---

#### 4. Lenke til Gitlab (Wiki, Issue Board og repo)

### Del 2: Individuell innlevering i Inspera

Individuelt refleksjonsnotat for hver deltaker i prosjektgruppen der en beskriver egen innsats og bidrag i prosjektet og hvordan en opplevde samarbeidet i prosjektgruppen. Dette dokumentet er individuelt og ikke synlig for de andre og skal inneholde følgende:

- Navn og scrum-team nummer.
- Beskriv team-prosessen slik du har opplevd den (hva gikk bra/ hva gikk mindre bra)
- Beskriv ditt bidrag og din rolle i prosjektet
- Vurdering og analyse av teamprosess og eget bidrag. - Hvorfor tror du teamprosessen ble slik du har beskrevet den i punkt 1 og 2.
- Se litt framover - basert på de erfaringer du har gjort deg i dette prosjektet dersom du kommer i en tilsvarende situasjon hvordan vil du håndtere den, hva har du lært gjennom dette prosjektet som du vil ta med deg videre
- Konklusjon - hva har du lært, er det noe fra dette prosjektet du ønsker å se nærmere på? Ditt personlige overordnet inntrykk av egen innsats, teamet, prosessen og prosjektet

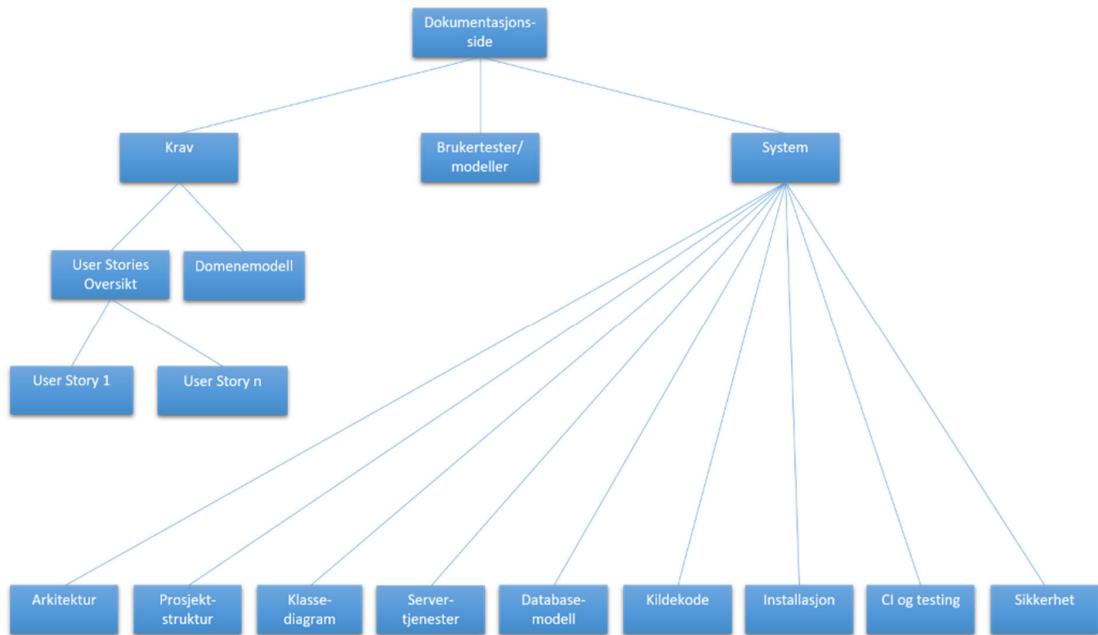
## 7 Vedlegg og sluttrapport

Sluttrapporten leveres som en PDF i henhold til mal og skal inneholde følgende:

- Sammendrag
- Forord
- Innholdsfortegnelse og evt. figur og tabelliste
- Oppgavetekst
- Introduksjon
- Teori og relevant litteratur
- Metode
- Resultater
  - Prosjektresultater
  - Administrative resultater
- Konklusjon og videre arbeider
- Referanser
- Vedlegg
  - Lenke til scrumteamets gitlab-repo
  - Lenke til scrumteamets gitlab-wiki
  - Lenke til scrumteamets gitlab-issueboard

- 
- Produktkø
  - Sprintkø sprint 1 og 2
  - Wiki (pdf av krav, brukertester/modeller og systemdokumentasjon)

## 7.1 WIKI-struktur og innhold



---

## GitLab lenker

Gitlab: [https://gitlab.stud.idi.ntnu.no/idatt2106\\_2024\\_05/idatt2106\\_2024\\_05](https://gitlab.stud.idi.ntnu.no/idatt2106_2024_05/idatt2106_2024_05)

Gitlab Wiki: [https://gitlab.stud.idi.ntnu.no/idatt2106\\_2024\\_05/idatt2106\\_2024\\_05/-/wikis/home](https://gitlab.stud.idi.ntnu.no/idatt2106_2024_05/idatt2106_2024_05/-/wikis/home)

---

## Møteinnkalling fra sprint 1 planlegging

---

Team 5  
GL-HB 415

15/04/2024  
12:00 - 13:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Agenda:

1. Presentere produkt backlog
2. Diskutere user stories
3. Eventuelt

Vel møtt!

Elias Trana

---

## Møtereferat fra sprint 1 planlegging

15/04/2024  
12:00 - 13:00

## Møtereferat

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Agenda:

1. Presentere produkt backlog
  - Satt oss inn i visjon, laget "initialisering" issues, begynt på wireframe etc.
  - Tilbakemelding:
    - Fokus på kun ett mål i starten, flere mål er eventuell implementasjon
    - Lag alle mål i starten, huk av underveis
2. Diskutere user stories
  - Aksjesparing - ikke prioriter
  - Felles sparemål, prioriter over budsjett
    - Ulike utfordringer per person, men inn på samme mål
    - Utfordre hverandre
    - Under oppretting, personlig / legg til medlemmer
  - Sparetips over nyheter?
  - Større fokus på sosialt
  - Nyheter og budsjett drøyes til sprint 2
  - Grisen skal vandre langs stien
  - Fokus på mobil, funksjonalitet for desktop
  - Feed ?
  - Oversiktsside
  - Dagens utfordring
  - Hjemmeside hvor du kan godkjenne
  - Nivåer på utfordringer (eks)
    - Nivå 1: Ikke kjøp snus på 1 uke

- 
- Nivå 2 (når klart nivå 1): Ikke kjøp snus på 3 uker
  - Lengde på utfordringer
  - Daglige utfordringer, går x skritt tilsvarende de daglige målene
  - Vise daglig sparing og total sparing
  - Felles sparemål:
    - Stien viser totalt spart blant alle
    - Oversikt over de individuelle til hverandre
  - Tracke progress, selv om du ikke klarer hele utfordringen
  - Hvis du mister streak:
    - Innskudd for å "betale" seg tilbake
    - Gjøre en vanskeligere utfordring for å komme seg tilbake
  - Etter du har gjort målene, animert visualisering av "bevegelsen"
  - Passe på at man kan godkjenne ting som allerede har skjedd, godkjenne gårdsdagen
  - Hvilke utfordringer passer for deg da du oppretter appen
  - Stien lages mens man setter opp utfordringer, når ikke frem hvis du ikke har nok utfordringer

### 3. Eventuelt

Vel møtt!

Elias Trana

---

**Møteinnkalling for møte med produkteier 19.04.2024**

---

Team 5  
GL-HB 415

19/04/2024  
12:00 - 13:00

## Møteinnkalling

Møteinnkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Agenda:

- Diskutere arbeid og prosess til nå
- Presentasjon av wireframes
- Tilbakemeldinger på nåværende løsninger

Vel møtt!

Elias Trana

---

**Møtereferat for møte med produkteier 19.04.2024**

---

Team 5  
GL-HB 415

19/04/2024  
12:00 - 13:00

## Møtereferat

Møteinnkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Agenda:

- Diskutere arbeid og prosess til nå
  - Ligger godt an
  - Har forstått produkteiers ønsker godt
- Presentasjon av wireframes
  - Fin implementasjon
  - Bra med fokus på mobilvisning
- Tilbakemeldinger på nåværende løsninger

Vel møtt!

Elias Trana

---

**Møteinkalling for sprint 1 review 25.04.2024**

---

Team 5  
GL-HB 415

25/04/2024  
12:00 - 13:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Sprint 1 review.

Agenda:

- Presentasjon av MVP
- Presentasjon av fremtidige planer
- Tilbakemeldinger fra produkteier
- Justeringer og forbedringer
- Fokus for Sprint 2

Vel møtt!

Elias Trana



---

Team 5  
GL-HB 415

25/04/2024  
12:00 - 13:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Sprint 1 review.

Agenda:

- Presentasjon av MVP
  - På fine sparemål, tom checkbox som blir fylt
  - Øye ikke inni på oppdater passord
  - Vis og endre utfordringer inni GamePath
- Presentasjon av fremtidige planer
  - Bilder
  - Bedre AI
- Tilbakemeldinger fra produkteier
  - Mer gamification
    - Badges
    - Achievements
  - Streak
  - Mer konkrete mål for sprinten
  - Sende stories for sprint 2 til PO etter dagens møte
- Justeringer og forbedringer
- Fokus for Sprint 2

Vel møtt!

Elias Trana



---

**Møteinnkalling for sprint 2 planlegging 25.04.2024**

---

Team 5  
GL-HB 415

25/04/2024  
10:30 - 11:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Spring 1 review.

Agenda:

- Tilbakemeldinger fra produkteier etter møte

Vel møtt!

Elias Trana

---

**Møtereferat for sprint 2 planlegging 25.04.2024**

---

Team 5  
GL-HB 415

25/04/2024  
10:30 - 11:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Spring 1 review.

Agenda:

- Tilbakemeldinger fra produkteier etter møte
  - Delte mål - høy
  - Badges & streak - høy
  - Dele mål og utfordringer i feed - høy
  - Profilside - høy
  - Fungerende bilder - høy
  - Vane-endringer - middels (både erfaren og hvor villig du er til å gjøre endringer)
    - Oppdatere informasjonen i entry-quiz på profil
  - Simulere opp mot konto - middels (husk å begrunne hvorfor vi ikke tar med dette i rapporten)
  - Total sti - lav (tydeliggjøre progressbar i pigpath)

Vel møtt!

Elias Trana

---

**Møteinkalling for sprint 2 review 03.05.2024**

---

Team 5  
GL-HB 415

03/05/2024  
14:00 - 15:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Sprint 2 review.

Agenda:

- Presentasjon av produkt
- Tilbakemeldinger
- Opprunning

Vel møtt!

Elias Trana

---

**Møtereferat for sprint 2 review 03.05.2024**

---

Team 5  
GL-HB 415

03/05/2024  
14:00 - 15:00

## Møteinkalling

Møteinkalling for:

Elias Trana, Emil Skogheim, Erik Nordsæther, Jacob Forsdahl Iqbal, Kristoffer Fredriksen, Sander Rom Skofsrud, Talian Stangebye-Hansen, Vegard Johnsen, Julie Bue Kulseth & Thea Marie Semundset Løseth

Vi ønsker dere velkommen til Sprint 2 review.

Agenda:

- Presentasjon av produkt
  - Små justeringer før innlevering
    - Små skrivefeil
    - Scroll til toppen
    - Profilbilde mangler i venneforerespørrelse
- Tilbakemeldinger
  - Godt gjennomført Sprint
  - God kommunikasjon
- Opprunning
  - Alle prioriterte user stories er oppnådd

Vel møtt!

Elias Trana

---

## Produkteiers prioriteringer

---

# Prioriteringer fra produkteier

1. Opprette mål
2. Tilpassede utfordringer
3. Visualisering og skalering (mobil-tilpassing)
4. Sosial side
5. Samarbeide om et mål med en annen bruker
6. Samle merker basert på prestasjon
7. Dele utfordringer og mål med andre brukere
8. Konfigurere og endre sparevaner
9. Simulere bankkonto

*Resten av prioriteringene og kravene følger prioriteringer i visjonsdokumentet*

## **Andre kommentarer og prioriteringer fra produkteier:**

Veldig viktig å fokusere på mobilt design, brukeropplevelse på alle enheter er høyt prioritert.

Sosiale aspekter er høyt prioritert, skal fungere like mye som et sosialt medium som en spare-applikasjon (uten å ofre noe av spare-funksjonaliteten)

AI-generering av utfordringer hadde vært et stort pluss

## **Akseptansekriterier:**

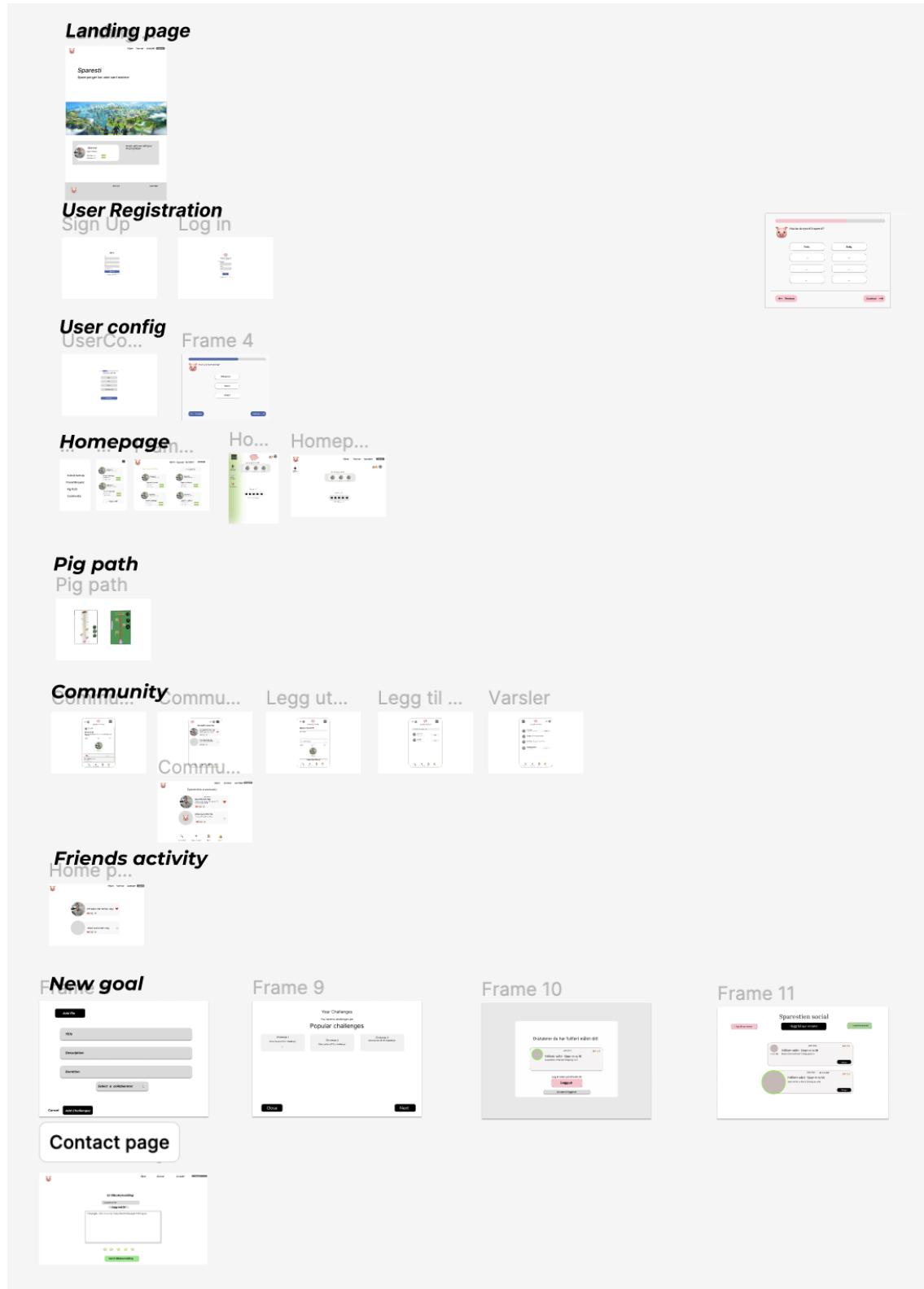
Produkteier sine kriterier for at brukerhistorier skal være fullført:

- **Brukerfunksjonalitet:** Brukere skal ha mulighet til å registrere en ny bruker, samt å logge inn og ut med brukeren. Ved registrering av bruker skal den bli tatt gjennom en konfigurering hvor den skal fylle ut erfaring, villighet til å endre vaner samt hvilke vaner den har lyst til å endre på.
- **Sparemål:** Brukere skal kunne opprette sine egne sparemål, med tilpassede utfordringer. Målet skal ha en målsum og måldato, samt navn og beskrivelse. Det er også ønskelig med bilder.
- **Spareutfordringer:** Brukere skal ha mulighet til å lage egne utfordringer, velge fra en forhåndsdefinert liste eller generere basert på brukerkonfigurasjonen. De forhåndsdefinerte eller genererte utfordringene skal også kunne redigeres for å passe brukerens behov best mulig.
- **Visualisering og utforming:** Programmet skal være enkelt og intuitivt i design. Designet skal være profesjonelt for å bygge troverdighet via det visuelle språket, men skal gjerne også inkludere mer lekne designelementer. Det må også fungere sømløst på mobil, da produkteier ser for seg applikasjonen mer som en mobil-app enn en web-applikasjon.
- **Sosial funksjonalitet:** En bruker skal ha mulighet til å legge til venner. Disse vennene skal ha tilgang til en feed hvor innlegg vises. Det skal også gjerne

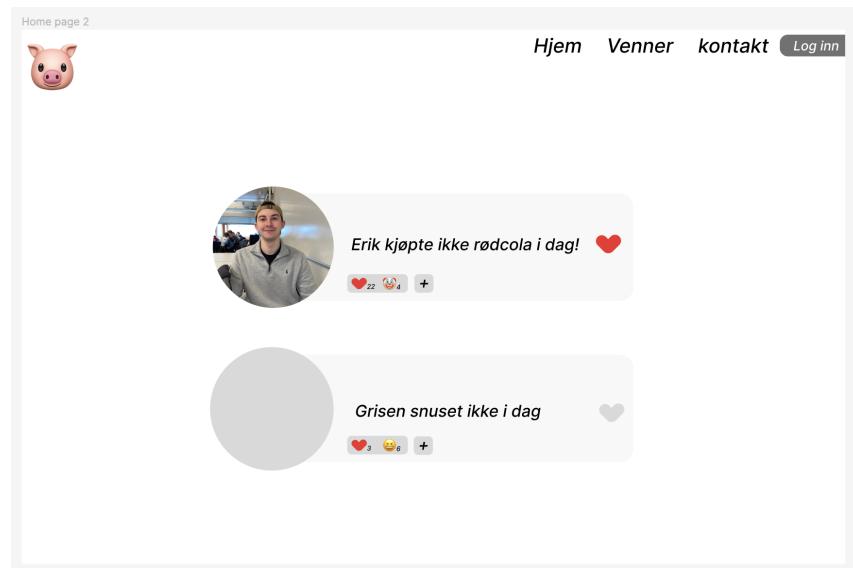
---

vises fullførte mål i feeden. Man skal også ha mulighet til å samarbeide på mål sammen med vennene sine. Det skal da være et felles mål, men med egne utfordringer tilpasset hver enkelt bruker.

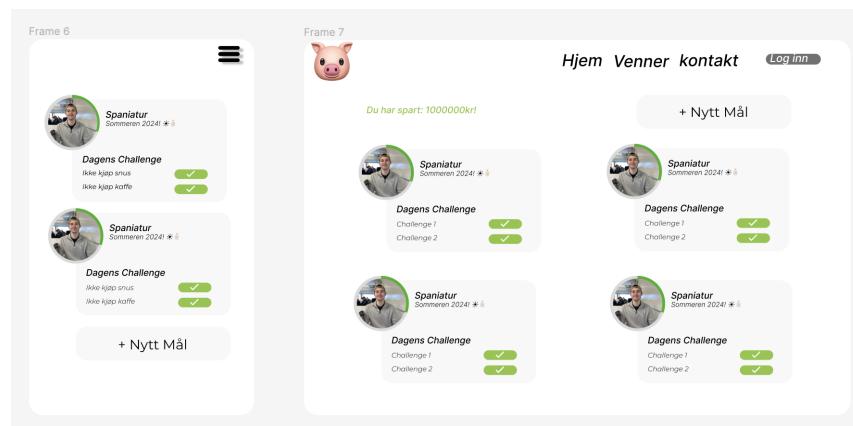
## Wireframe



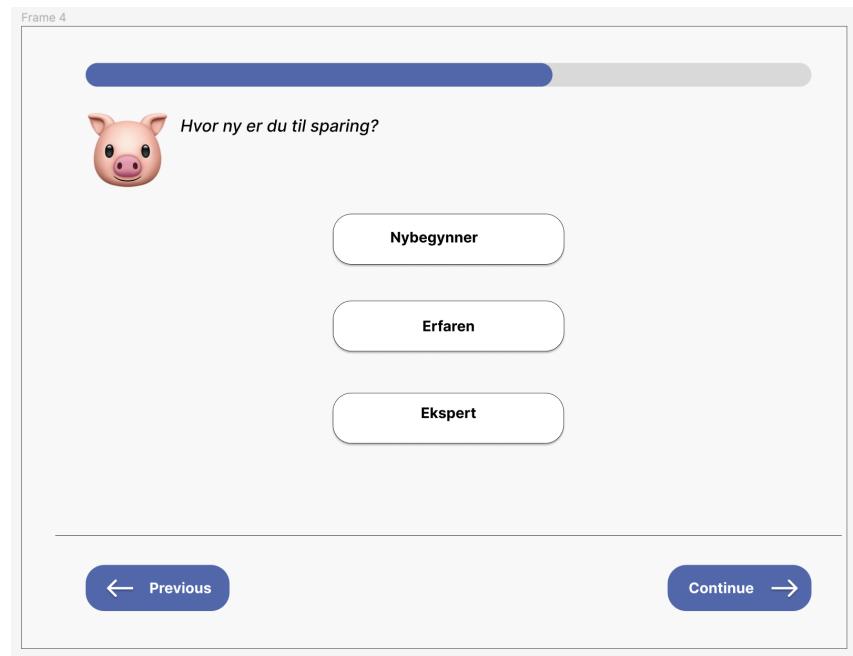
Figur 5: Wireframe



Figur 6: Wireframe - Sosial



Figur 7: Wireframe - Mål



Figur 8: Wireframe - Konfigurerering

---

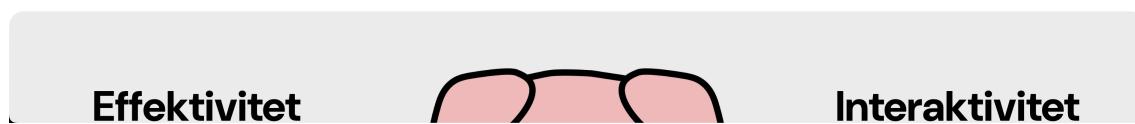
## Visuell profil



# SPARESTI

Veien til dine mål - et steg av gangen

Start



Figur 9: Endelig design - Forsiden av applikasjonen



Figur 10: Endelig design - Applikasjonens egenskaper



**Logg inn**



Brukernavn

Passord

**Logg inn**

[Glemt passord?](#)

Har du ikke en bruker? [Lag bruker](#)

Figur 11: Endelig design - Logg inn visningen

---

## Usertest 1

---

## **Brukertest 1, 26.04.2024**

### **Hva er formålet med testen?**

Formålet med testen var å teste MVP-en for å få tilbakemeldinger på hva vi må jobbe med og fokusere på opp mot den endelige versjonen

### **Hvilken funksjonalitet skal testes?**

Programmet skal i sin helhet testes. Gir ting mening, føles programmet naturlig å bruke?

### **Hjem utfører testen?**

Pedro Pablo Cardona Arroyave, veileder og studentassistent i faget IDATT2106

### **Hvor finner testen sted?**

Testen vi finne sted på GL-HB 415

### **Utstyr**

Deltakeren vil få tilgang til en PC med programmet kjørende

### **Oppgaver brukeren skal gjennomføre**

1. Opprette bruker og logge inn
2. Opprette et sparemål med utfordringer
3. Gjennomføre en utfordring
4. Interagere med og utforske "sparestien"
5. Utforske programmet i sin helhet

### **Spørsmål stilt etter testen**

- **Hva var førsteinntrykket ditt av applikasjonen?**
  - Veldig bra med forsiden og informasjonen der
  - Nyttig med om oss siden for å presentere teammedlemmer og tilbakemeldinger
  - Generelt sett rent og ryddig design
- **Har du noen generelle forslag på ting som bør endres?**
  - Login knappen bør ha større kontrast
  - Grisen på forsiden kræsjer litt med det seriøse og rene designet
  - Pass på å skille mellom feedback og report, ettersom disse er forskjellige typer tilbakemeldinger
  - **Challenges komponenten:**

- 
- Legge til valutasymbol på utfordringer for å tydeliggjøre hva det omhandler
  - Tydeliggjøre hvis et mål ikke har noen aktive eller tilknyttede utfordringer
  - Sørge for at fremdriftslinjen i komponenten viser riktig for totalt spart
- **Sosial-siden:**
- Bør ikke vise en oversikt over alle brukere ved valg av “add friend”
  - Pass på å ha paginering for innlegg og i “add friends” seksjonen

## **Testgruppe**

Elias og Sander vil ha ansvar for gjennomføringen av brukertesten.

---

## Usertest 2

---

## **Brukertest 2, 02.05.2024**

### **Hva er formålet med testen?**

Formålet med testen var å teste et nesten ferdig produkt for å få tilbakemeldinger på hva vi må jobbe med og fikse rett opp mot innlevering

### **Hvilken funksjonalitet skal testes?**

Programmet skal i sin helhet testes. Gir ting mening, føles programmet naturlig å bruke? Det vil også være et større fokus på den sosiale funksjonaliteten, legge til venner og opprette delte mål.

### **Hvem utfører testen?**

Pedro Pablo Cardona Arroyave, veileder og studentassistent i faget IDATT2106

### **Hvor finner testen sted?**

Testen vi finne sted på GL-HB 415

### **Utstyr**

Deltakeren vil få tilgang til en PC med programmet kjørende

### **Oppgaver brukeren skal gjennomføre**

1. Opprette bruker og logge inn
2. Opprette et sparemål med utfordringer
3. Konfigurere bruker, profilbilde
4. Gjennomføre en utfordring
5. Interagere og utforske "sparestien"
6. Legge til venner
7. Opprette og gjennomføre delte mål
8. Utforske programmet i sin helhet

### **Spørsmål stilt etter testen**

- **Hvordan opplevdes de sosiale aspektene ved programmet?**

- Bør vise varsling ved venneforerespørsel
- Må vise bilder på venner og forespørsler
- Vise personene du samarbeider på et mål med
- Bra med individuelle utfordringer på delte mål
- Bra å vise individuelle bidrag på delte mål

- 
- **Har du noen generelle forslag på ting som bør endres?**
    - Linker på nyheter bør ikke være blå
    - Fjerne alertbox fra feedback
    - Tydeliggjøre 2FA epost med varighet, gjøre den litt mer troverdig
    - Utfordringer bør vises automatisk ved opprettet mål
    - Bør vise mer enn bare 5-stjerners anmeldelser
    - Legg til knapp for å fjerne venner
    - Gi brukeren tilbakemelding etter fullført undersøkelse

### **Testgruppe**

Elias og Sander vil ha ansvar for gjennomføringen av brukertesten.

---

## Produktkø

## Produktkø

Behov	Beskrivelse	Prioritet
Lage sparemål	<p>Bruker skal kunne opprette et eller flere sparemål, der de setter opp</p> <ul style="list-style-type: none"> <li>- Navn på sparemål</li> <li>- Hvor mye som skal spares</li> <li>- Når målet skal være nådd</li> <li>- Bilde</li> </ul>	Høy
Motta persontilpassede utfordringer	<p>En bruker skal kunne lage egne utfordringer som inneholder:</p> <ul style="list-style-type: none"> <li>- Navn</li> <li>- Lengde</li> <li>- Potensiell spare mengde</li> </ul> <p>Det skal også være mulig å generere utfordringer ved bruk av KI eller velge fra en liste med populære utfordringer</p>	Høy
Oversikt over penger spart	<ul style="list-style-type: none"> <li>- Se totalt spart ved bruk av appen</li> <li>- Se totalt spart per sparemål</li> <li>- Visuell oppsummering av status på sparemål</li> </ul>	Høy
Enkelt brukergrensesnitt / Visualisering	<p>Brukeren skal enkelt finne frem til funksjonalitetene den ønsker. Appen skal være oversiktlig og intuitiv. Den skal også inneholde grafiske elementer for å vise blant annet prosesjon i mål og utfordringer. Alt beskrevet over må også fungere på mobil</p>	Høy
Sosial side	<p>Brukeren skal kunne legge til venner. Den skal også kunne legge ut innlegg for vennene sine, og se venner sine innlegg. Ved fullført mål skal dette automatisk legges ut.</p>	Medium

Badges for motivasjon og følelse av oppnåelse	En bruker skal kunne oppnå medaljer (badges) basert på totale penger spart. Den skal også få en streak som oppdateres ved daglig oppnåelse av utfordringer	Medium
Sparing mot felles mål	Brukere kan lage et sparemål sammen med andre brukere, og man kan se hvor mye en partner venn har spart mot samme mål. Et delt mål skal ha personlige utfordringer tilpasset hver enkelt bruker.	Medium
Simulere konto	Programmet skal ha funksjonalitet for å simulere oppkoblinger mot en bankkonto.	Lav
Dele utfordringer	En bruker skal kunne dele utfordringene sine med andre brukere.	Lav

---

## Sprintkø Sprint 1

## Sprint-backlog 1

Issue	Estimated time in hours	Status
Mobile view for add goal	1	Done
Update mobile scaling	1	Done
Set up GitLab	1	Done
Set up milestones	0.167	Done
Create EER-diagram	0.5	Done
User handling	3	Done
Fix mobile scale on footer	0.5	Done
Update styling for contact	1.5	Done
Set up labels	0.167	Done
Create auth, user DTO and mapper	1	Done
Initialize springboot	0.167	Done
Add JWT token generation and security config	0.5	Done
Regex login/create	2	Done
Fix mobile navbar	1.5	Done
set up router	0.167	Done
Init UUE	0.5	Done
Refactor shared goals and goal	2	Done
create user config	2	Done
implement goal creation	2	Done
service layer	6	Done
Create private and public user controller	0.5	Done
Implement Global exception handler	1	Done
update User info	0.5	Done
create navbar and footer	3	Done
Friend request functionality	2	Done

Create pig path	10	Done
Create goal wheel component	0.5	Done
Create goal and usergoal controller	1	Done
Create goal and usergoal DTO and mapper	4	Done
Create Goal and usergoal entity	3	Done
Fix picture in feedback	1	Done
Add swagger	Con	Done
Implement cooperation in goals (shared goals)	2	Done
Economic Habit entity	2	Done
Create challenge and sub-challenge entity	2	Done
Create login page	1	Done
Create post/ fix font and button	0.5	Done
Add friend	3	Done
UserEconomic habit entity	2	Done
Axios setup	5	Done
Create Axios for token handling	1	Done
create challenge and sub-challenge controller	1	Done
Change navbar	0.5	Done
Display goals	4	Done
Create Challenge and sub-challenge DTO and mapper	3	Done
Implement friendship	2	Done
Create landing page	3	Done
Goal component	2	Done
Fix challenge creation	6	Done
Add profile page	3	Done
Create Signup page	3	Done
Create Homepage	4	Done
Reaction service for post	1.5	Done

Router from signup to login	0.5	Done
Show daily challenge	4	Done
Endpoint for complete goal	3	Done
Create user frontend	0.75	Done
POST service	1.5	Done
Create docker compose for building	0.167	Done
Create user config page	3	Done
Check challenges	2	Done
Create friend request component	1	Done
Refactor packaging	0.25	Done
Wireframe	8	Done
User role implementation	0.75	Done
Feedback / contact	4	Done
User handling	2	Done
Add friends to profile page	1	Done
Mock pig game	6	Done
Create templates	2	Done
Add sas to User goal	2	Done
Add friends	3	Done
2FA	5	Done
Start date goal	0.5	Done
Login logic	0.3	Done
Fix backend dependencies	0.5	Done
User test	2	Done
Update user info	0.5	Done
Community startpage	5	Done
Challenge	2	Done
Progress daily challenge	2	Done
Friendship store	2	Done

---

Display friends	4	Done
Automatic shared goal	0.5	Done
Scroll to top switch	0.5	Done
Show daily in pigpath	6	Done
Goals feed	7	Done
Show username on register	0.67	In progress
Need Refresh on Goal-create	1	In progress
Forgot password	2	In progress
Implement grafana	4	Backlog
User statistic	2	Backlog
Handle duplicated challenges	4	Backlog
implement Challenge templates	2	Backlog
Forgot password page	1	Backlog
Config NTNU database for prod build	0.5	Backlog
News page	2	Backlog
Remote database	0.5	Backlog
CI/CD pipeline	1	Review
Properties files	1	Review
Create service for endpoints	5	Review
Prompt engineering	6	Review

---

## Sprintkø Sprint 2

---

## Sprint-backlog 2

Issue	Estimated time in hours	Status
Fix popular challenges	0.5	Done
Expand global exception handler	2	Done
Completion of monetary challenges	6	Done
Experience in user config	0.5	Done
View progress of contributors	3	Done
Update username	1	Done
Update navbar	2	Done
Show badges in profile	1.5	Done
Entity for badges	3	Done
Fix bug with friend request	1	Done
Role api layer tests	1	Done
Social api layer tests	1	Done
Refactor post component	1	Done
Complete goal pop up	2	Done
Show badges in feed	1.5	Done
Display error message	2	Done
Shared goal view	1	Done
Create shared goal	1.5	Done
Streak component	1	Done
Track total saving	6	Done
Refine pathgame	0.5	Done
Fix friends	2	Done
Resolve reactivity issues	3	Done
Total progress in game view	1.5	Done

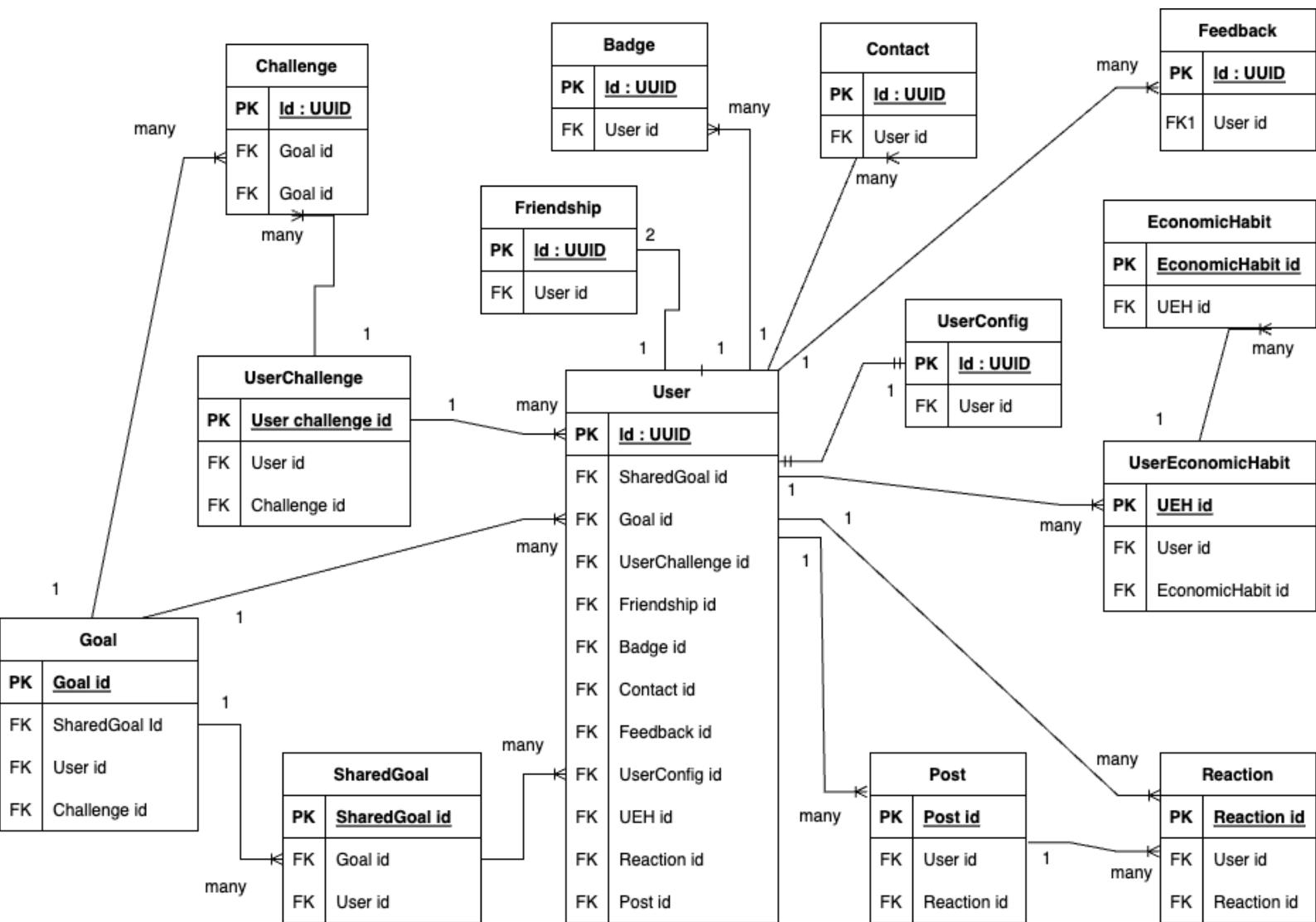
Implement streak in user	2	Done
View badge progress	3	Done
Show other profiles	1	Done
Update user survey	0.5	Done
Update profile picture	3	Done
Add friend, display with search	1.5	Done
Show info based on userID	0.5	Done
Update streak in user	3	Done
Refactor economic habit	2	Done
Show friends in profile	1	Done
Edit challenge for goals	2	Done
Persistence layer tests	2	Done
Economic API-layer tests	1	Done
Personal challenge shared goal	6	Done
View contributors in shared goal	4	Done
Commitment and selected habit	Extension of Prompt engineering from sprint 1 (6)	Done
Update config details	2	Done
Add picture to goal	3	Done
User API-layer tests	1	Done
Friends in feed	1	Done
Change username or password	0.5	Done
Write unit tests	10	Done
Role service layer tests	1	Done
Shared API - layer tests	1	Done
Goal API - Layer tests	1	Done
Challenge API-layer tests	1.5	Done

---

Feedback API-layer tests	1	Done
E2E tests	10	Done
Clear all on log out for other users goals	0.167	Done
Only one challenge shown on creation	0.0835	Done
Shows error message from backend	1	Done
Add validation	3	Done
Azure service layer tests	1	Done
Frontend share challenge on creation	3	Backlog
Show previous challenges	5	Backlog
Add admin panel	5	Backlog
Use store, not service friendships	2	Backlog

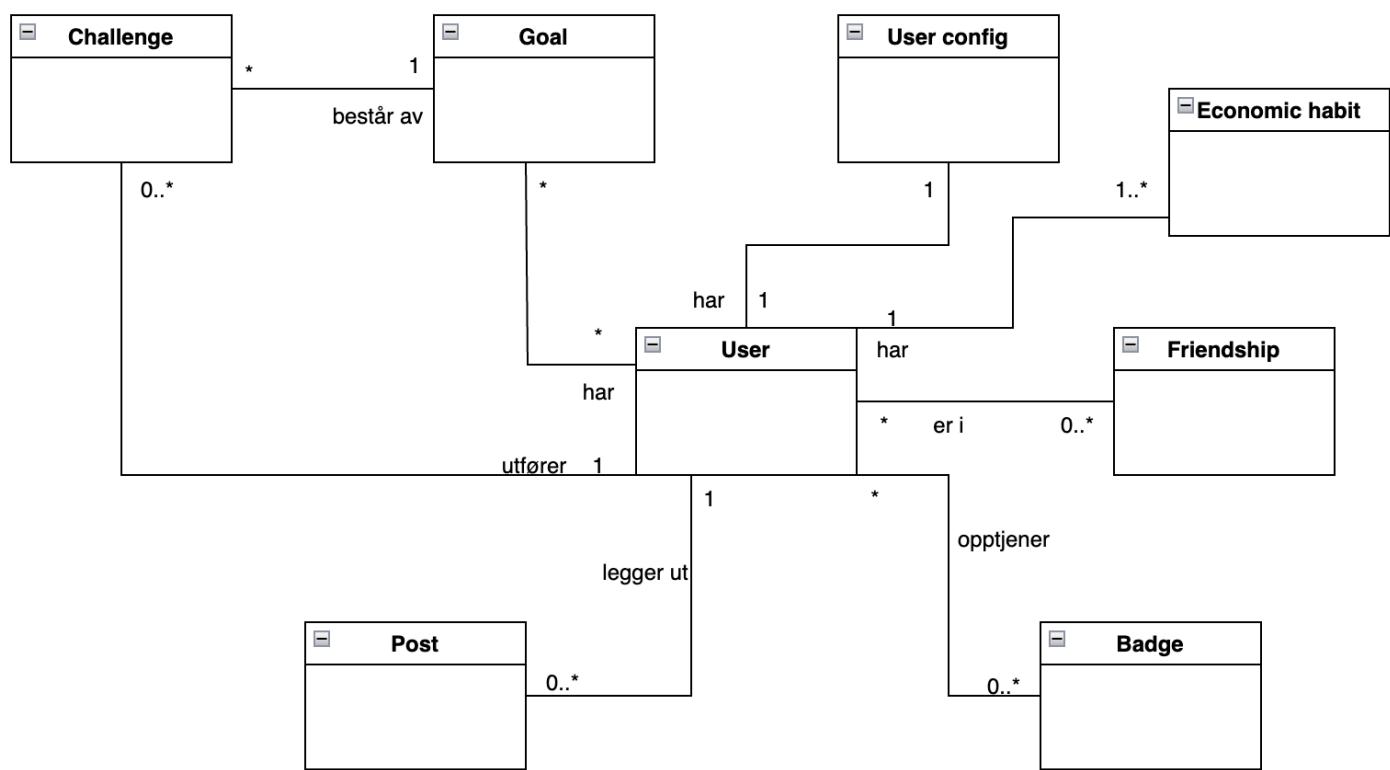
---

**ER diagram fra 25.04**



---

## Domenemodell



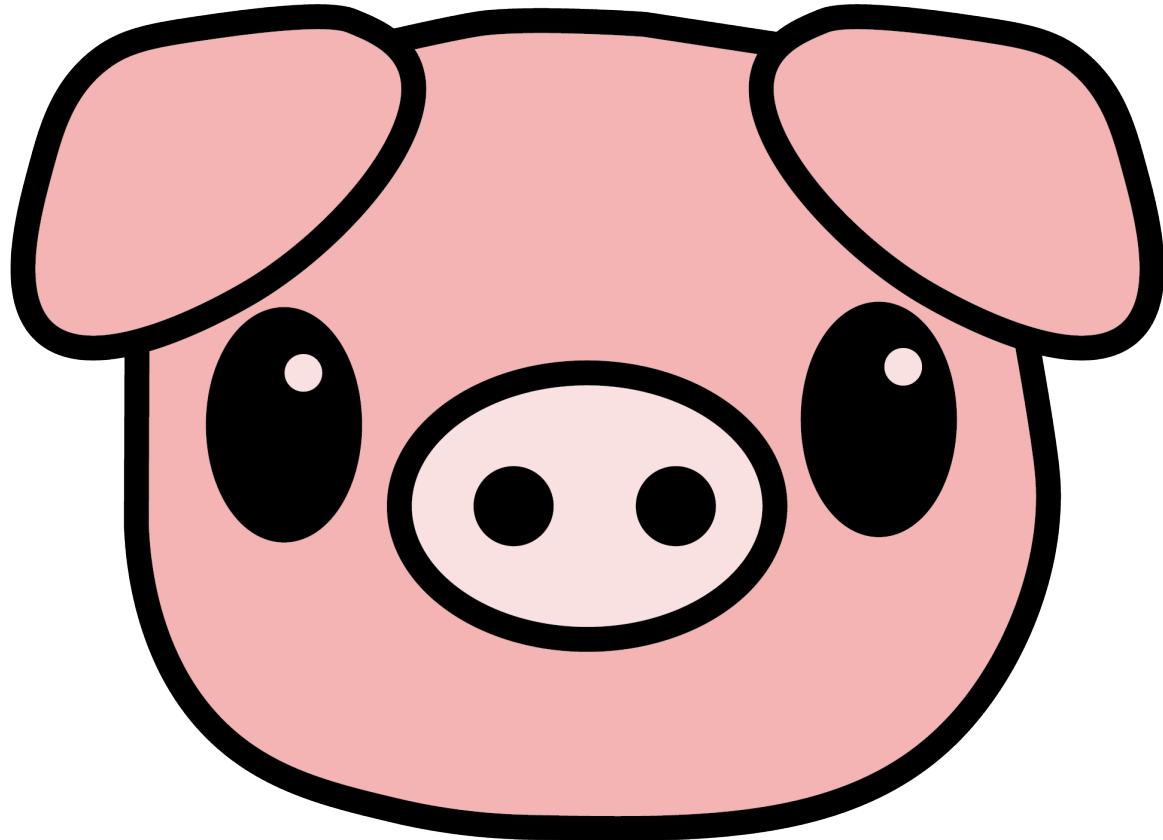
---

Wiki

---

Last edited by  [Lars Talian Stangebye-Hansen](#) 1 day ago

## Home



# SPARESTI

Dette er en samlet wiki for både frontend og backend i faget IDATT2106 på NTNU Trondheim. Laget av Scrum team 5:

Emil Skogheim

Lars Talian Stangebye-Hansen

Kristoffer Fredriksen

Jacob Forsdahl Iqbal

Elias Trana

Sander Rom Skofsrud

Vegard Johnsen

Erik Nordsæther

Krav	User Stories Oversikt	System
<a href="#">Brukertester</a>	<a href="#">Registrere bruker</a>	<a href="#">Arkitektur</a> W2
<a href="#">Domenemodell</a>	<a href="#">Konfigurering</a>	<a href="#">Prosjektstruktur</a>

---

	<a href="#">Login</a>	<a href="#">Klassediagram</a>
	<a href="#">Feed for nyheter og sparetips</a>	<a href="#">Servingtjenester</a>
	<a href="#">Sparemål</a>	<a href="#">Databasemodell</a>
	<a href="#">Generere spareutfordringer</a>	<a href="#">Kildekode</a>
	<a href="#">Spareoversikt</a>	<a href="#">Installasjon</a>
	<a href="#">Spareutfordring</a>	<a href="#">CI og testing</a>
	<a href="#">Gamification</a>	<a href="#">Sikkerhet</a>
	<a href="#">Profil</a>	
	<a href="#">Mobil tilpassing</a>	
	<a href="#">Sosial side</a>	
	<a href="#">Delt mål</a>	

---

Last edited by  [Sander Rom Skofsrud](#) 9 hours ago

# Brukertesting

## Brukertest 1, 26.04.2024

### Hva er formålet med testen?

Formålet med testen var å teste MVP-en for å få tilbakemeldinger på hva vi må jobbe med og fokusere på opp mot den endelige versjonen

### Hvilken funksjonalitet skal testes?

Programmet skal i sin helhet testes. Gir ting mening, føles programmet naturlig å bruke?

### Hvem utfører testen?

Pedro Pablo Cardona Arroyave, veileder og studentassistent i faget IDATT2106

### Hvor finner testen sted?

Testen vi finne sted på GL-HB 415

### Utstyr

Deltakeren vil få tilgang til en PC med programmet kjørende

### Oppgaver brukeren skal gjennomføre

1. Opprette bruker og logge inn
2. Opprette et sparemål med utfordringer
3. Gjennomføre en utfordring
4. Interagere med og utforske "sparestien"
5. Utforske programmet i sin helhet

### Spørsmål stilt etter testen

- **Hva var førsteinntrykket ditt av applikasjonen?**
  - Veldig bra med forsiden og informasjonen der
  - Nyttig med om oss siden for å presentere teammedlemmer og tilbakemeldinger
  - Generelt sett rent og ryddig design
- **Har du noen generelle forslag på ting som bør endres?**
  - Login knappen bør ha større kontrast
  - Grisen på forsiden kræsjer litt med det seriøse og rene designet
  - Pass på å skille mellom feedback og report, ettersom disse er forskjellige typer tilbakemeldinger
  - **Challenges komponenten:**
    - Legge til valutasymbol på utfordringer for å tydeliggjøre hva det omhandler
    - Tydeliggjøre hvis et mål ikke har noen aktive eller tilknyttede utfordringer
    - Sørge for at fremdriftslinjen i komponenten viser riktig for totalt spart
- **Sosial-siden:**
  - Bør ikke vise en oversikt over alle brukere ved valg av "add friend"
  - Pass på å ha paginering for innlegg og i "add friends" seksjonen

### Testgruppe

Elias og Sander vil ha ansvar for gjennomføringen av brukertesten.

## Brukertest 2, 02.05.2024

### Hva er formålet med testen?

Formålet med testen var å teste et nesten ferdig produkt for å få tilbakemeldinger på hva vi må jobbe med og fikse rett opp mot innlevering

### Hvilken funksjonalitet skal testes?

Programmet skal i sin helhet testes. Gir ting mening, føles programmet naturlig å bruke? Det vil også være et større fokus på den sosiale funksjonaliteten, legge til venner og opprette delte mål.

### Hvem utfører testen?

W4

Pedro Pablo Cardona Arroyave, veileder og studentassistent i faget IDATT2106 **Hvor finner testen sted?**

## Utsyn

Deltakeren vil få tilgang til en PC med programmet kjørende

### Oppgaver brukeren skal gjennomføre

1. Opprette bruker og logge inn
2. Opprette et sparemål med utfordringer
3. Konfigurere bruker, profilbilde
4. Gjennomføre en utfordring
5. Interagere og utforske "sparestien"
6. Legge til venner
7. Opprette og gjennomføre delte mål
8. Utforske programmet i sin helhet

### Spørsmål stilt etter testen

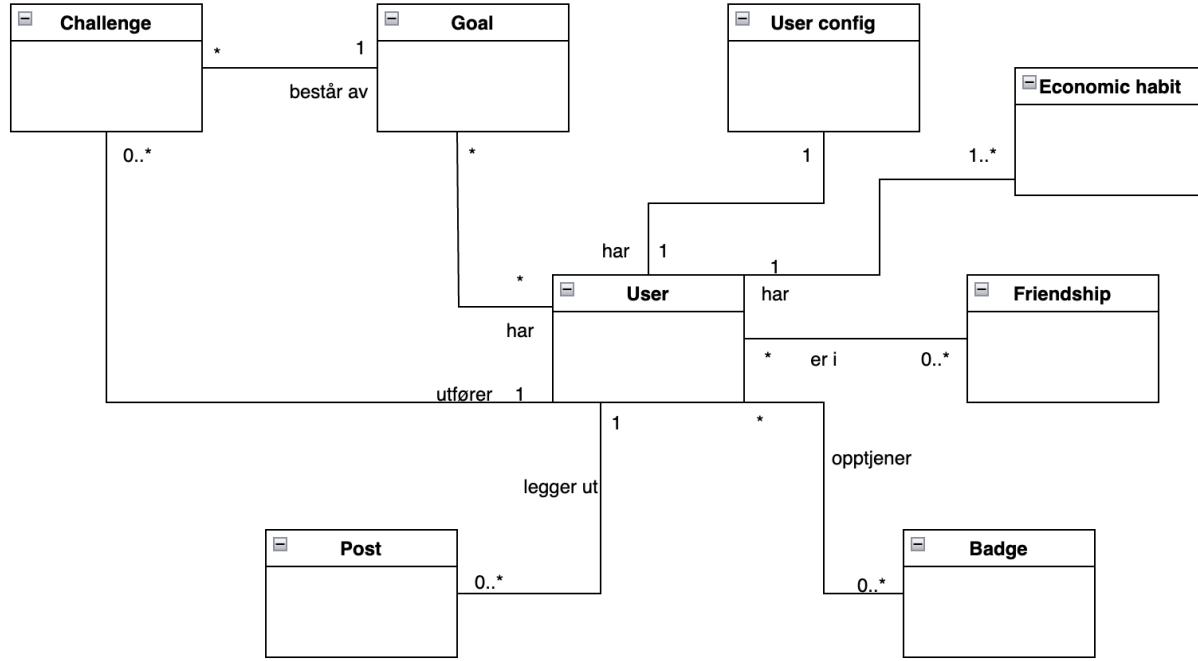
- Hvordan opplevdes de sosiale aspektene ved programmet?
  - Bør vise varsling ved venneforerespørsel
  - Må vise bilder på venner og forespørsler
  - Vise personene du samarbeider på et mål med
  - Bra med individuelle utfordringer på delte mål
  - Bra å vise individuelle bidrag på delte mål
- Har du noen generelle forslag på ting som bør endres?
  - Linker på nyheter bør ikke være blå
  - Fjerne alertbox fra feedback
  - Tydeliggjøre 2FA epost med varighet, gjøre den litt mer troverdig
  - Utfordringer bør vises automatisk ved opprettet mål
  - Bør vise mer enn bare 5-stjerners anmeldelser
  - Legg til knapp for å fjerne venner
  - Gi brukeren tilbakemelding etter fullført undersøkelse

### Testgruppe

Elias og Sander vil ha ansvar for gjennomføringen av brukertesten.

Last edited by  [Kristoffer Thorsdal Fredriksen](#) 1 hour ago

## Domenemodell



Last edited by  [Emil Skogheim](#) 5 hours ago

## Registrere bruker

### User story

Som: Potensiell ny bruker

Ønsker jeg: Å registrere meg

Slik at: Jeg kan opprette en konto og begynne å bruke applikasjonen

**\*\*Akseptansekriterier \*\***

- Brukeren skal kunne registrere seg ved å fylle ut nødvendig personlig informasjon.
- Brukeren skal motta en bekreftelse på vellykket registrering.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

# Konfigurering

## User story

Som: Bruker

Ønsker jeg: Å konfigurere mine sparepreferanser ved første pålogging

Slik at: Appen kan tilby tilpassede spareutfordringer basert på mine preferanser og vaner

## Akseptansekriterier

- Under første pålogging, skal brukeren kunne angi sin kjennskap til sparing, vaneendringer de er villige til å gjøre, og hvilke spareutfordringer de kunne tenke seg.
- Appen skal bruke denne informasjonen til å tilpasse utfordringer og tips til brukeren.

Last edited by  [Emil Skogheim](#) 5 hours ago

# Login

## User story

Som: Bruker

Ønsker jeg: Å logge inn i appen

Slik at: Jeg kan få tilgang til mine personlige spareressurser og informasjon

## Akseptansekkriterier

- Brukeren skal kunne logge inn med valgte autentiseringsmetoder (passord, fingeravtrykk, FaceID eller BankID).
- Systemet skal gi tilbakemelding ved feil innlogging.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

## Feed for nyheter og sparetips

### User story

Som: Bruker

Ønsker jeg: Å se en feed med nyheter og sparetips

Slik at: Jeg kan holde meg oppdatert på relevante økonomiske nyheter og få tips til hvordan spare mer effektivt

### Akseptansekriterier

- Feed skal oppdateres regelmessig med relevant og nyttig innhold.
- Brukeren skal kunne tilpasse hva slags tips og nyheter som vises basert på egne preferanser.

Last edited by  [Emil Skogheim](#) 5 hours ago

# Sparemål

## User story

Som: Bruker

Ønsker jeg: Å sette personlige sparemål

Slik at: Jeg kan ha klare mål for min sparing

## Akseptansekrriteria

- Brukeren skal kunne opprette, redigere og slette sparemål.
- Appen skal vise fremdrift på hvert sparemål.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

## Generere spareutfordringer

### User story

Som: Bruker

Ønsker jeg: At appen skal generere spareutfordringer for meg

Slik at: Jeg kan bli utfordret til å spare mer gjennom konkret og målbar aktivitet

### Akseptansekriterier

- Spareutfordringene skal være basert på brukerens tidligere oppgitte preferanser og sparehistorikk.
- Brukeren skal kunne akseptere eller avslå hver utfordring.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

## Spareoversikt

### User story

Som: Bruker

Ønsker jeg: Å se en oversikt over mine spareaktiviteter

Slik at: Jeg kan følge med på fremgang og planlegge videre sparing

### Akseptansekkriterier

- Oversikten skal inkludere alle aktive og fullførte sparemål samt utfordringer.
- Appen skal vise detaljert fremgang for hvert mål og utfordring.

Last edited by  [Emil Skogheim](#) 5 hours ago

# Spareutfordring

## User story

Som: Bruker

Ønsker jeg: Å gå inn på spesifikke spareutfordringer i min spareoversikt

Slik at: Jeg kan få detaljert informasjon om utfordringen og se min prosesjon

## Akseptansekriterier

- Detaljert visning av hver utfordring inkludert forventet sparebeløp og gjenværende aktiviteter.
- Mulighet for manuell registrering av fullført sparing knyttet til utfordringen.

Last edited by  [Emil Skogheim](#) 5 hours ago

# Gamification

## User story

Som: Bruker

Ønsker jeg: Å oppleve elementer av gamification mens jeg sparer

Slik at: Sparing blir mer engasjerende og motiverende

## Akseptansekriterier

- Systemet skal tilby badges og streaks som belønning for gjennomførte utfordringer.
- Progressbærer skal vise brukerens fremgang i utfordringene.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

# Profil

## User story

Som: Bruker

Ønsker jeg: Å administrere min personlige profil

Slik at: Jeg kan oppdatere mine personopplysninger, kontovalg og se mine prestasjoner

## Akseptansekriterier

- Brukeren skal kunne oppdatere personlige opplysninger og innstillinger.
- Profilen skal vise oversikt over oppnådde badges og andre prestasjoner.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

# Mobil tilpassing

## User story

Som: Bruker

Ønsker jeg: At applikasjonen skal være visuelt tiltalende og funksjonell på mobil

Slik at: Jeg kan bruke applikasjonen effektivt på forskjellige enheter

## Akseptansekriterier

- Designet må være intuitivt og profesjonelt, med lekne elementer som passer til et bredt publikum.
- Applikasjonen må fungere sømløst og være optimalisert for mobilbruk.

---

Last edited by  [Emil Skogheim](#) 5 hours ago

## Sosial side

### User story

Som: Bruker

Ønsker jeg: Å bruke applikasjonen som et sosialt medium

Slik at: Jeg kan dele og samarbeide om sparemål med venner

### Akseptansekriterier

- Brukere skal kunne legge til venner og se en feed av innlegg og fullførte mål.
- Brukere skal kunne samarbeide om felles sparemål, med tilpassede utfordringer for hver deltaker.

Last edited by  [Emil Skogheim](#) 5 hours ago

## Delt mål

### User story

Som: Bruker

Ønsker jeg: Å samarbeide om sparemål med venner

Slik at: Vi kan motivere hverandre og nå våre sparemål sammen

### Akseptansekriterier

- Det skal være mulig å opprette et felles sparemål med en eller flere venner.
- Hver deltaker skal kunne ha egne tilpassede utfordringer knyttet til det felles målet.

---

Last edited by  [Lars Talian Stangebye-Hansen](#) 6 hours ago

# Arkitektur

Sparesti-applikasjonen er bygget med en moderne fullstack-arkitektur som sikrer robusthet, skalerbarhet og fleksibilitet. Denne arkitekturen består av flere lag og komponenter som samarbeider sømløst for å tilby en sikker og effektiv brukeropplevelse. Her er en detaljert oversikt over de ulike komponentene og hvordan de er strukturert.

## Oversikt over arkitekturen

**Frontend:** Sparestis brukergrensesnitt er utviklet med Vue.js, et progressivt JavaScript-rammeverk kjent for sin fleksibilitet og ytelse. Frontenden er ansvarlig for å presentere data og funksjonalitet til brukerne på en intuitiv og responsiv måte.

**Backend:** Backenden er bygget med Spring Boot, et kraftig rammeverk for Java-baserte applikasjoner. Den håndterer forretningslogikk, databehandling, autentisering og autorisasjon, samt kommunikasjon med databasen og andre eksterne tjenester.

**Database:** Sparesti bruker MySQL, en pålitelig og skalerbar relasjonsdatabase, for å lagre all vedvarende data, inkludert brukerdata, transaksjoner og metadata.

**Containerization:** Alle komponentene kjører i Docker-containere, som sikrer miljøkonsistens, enkel distribusjon og skalerbarhet. Docker orkestreres ved hjelp av Docker Compose for å håndtere avhengigheter og tjenesteoppstart.

**Cloud Storage:** Bilder og andre mediefiler lagres i Azure Blob Storage, en skybasert lagringstjeneste som gir høy tilgjengelighet, skalerbarhet og sikkerhet.

## Frontend

**Vue.js:** Frontenden er bygget med Vue.js, som muliggjør rask utvikling og enkel komponentbasert arkitektur. Brukergrensesnittet er designet for å være responsivt og interaktivt, og gir en sømløs opplevelse på tvers av enheter.

**State Management:** Pinia brukes til state management, noe som sikrer konsistent datahåndtering og tilstand gjennom hele applikasjonen.

**Routing:** Vue Router administrerer klient-side routing, slik at applikasjonen kan navigere mellom forskjellige visninger uten å måtte laste siden på nytt.

## Backend

**Spring Boot:** Backenden benytter Spring Boot for å håndtere alle server-side operasjoner. Det inkluderer:

- **REST API:** Spring Boot brukes til å utvikle RESTful API-er som frontenden kommuniserer med.
- **Sikkerhet:** Spring Security integreres for autentisering og autorisasjon, inkludert implementasjon av to-faktor autentisering (2FA).
- **Datatilkobling:** Spring Data JPA brukes for databaseinteraksjoner, som gjør det enkelt å utføre CRUD-operasjoner på MySQL-databasen.

**JWT Tokens:** For sesjonshåndtering og sikkerhet bruker backenden JSON Web Tokens (JWT), som lagres i cookies på klienten. JWT sikrer at brukerens autentiseringstoken automatisk slettes når nettleserøkten avsluttes.

## Database

**MySQL:** Databasen lagrer all vedvarende informasjon, se her. [Databasemodell](#)

## Containerization

**Docker:** Alle applikasjonskomponenter kjører i Docker-containere, som sikrer at de kjører i isolerte miljøer. Dette gir flere fordeler, inkludert:

- **Isolasjon:** Hver komponent kjører uavhengig, noe som forhindrer konflikter og gjør det lettere å feilsøke.
- **Skalerbarhet:** Containere kan enkelt skaleres opp eller ned basert på belastning.
- **Portabilitet:** Docker-containere kan kjøres på ethvert miljø som støtter Docker, noe som gjør det enkelt å distribuere applikasjonen på tvers av forskjellige plattformer.

**Docker Compose:** Brukes til å administrere flercontainer-applikasjoner, inkludert definisjon av tjenester, nettverk og volumer som kreves for å kjøre applikasjonen.

## Cloud Storage

**Azure Blob Storage:** For lagring av bilder og andre mediefiler bruker Sparesti Azure Blob Storage. Dette gir flere fordeler:

- **Sikkerhet:** Bruk av SAS-tokens (Shared Access Signature) for å begrense tilgang til filene.
- **Skalerbarhet:** Ubegrenset lagringskapasitet for å håndtere veksten av data
- **Høy tilgjengelighet:** Sikrer at data alltid er tilgjengelig for applikasjonen

---

Sparesti-applikasjonens arkitektur er designet for å være robust, skalbar og sikker. Gjennom bruk av moderne teknologier som Vue.js, Spring Boot, MySQL, Docker og Azure Blob Storage, sikrer vi en sømløs og pålitelig brukeropplevelse. Vår containerbaserte tilnærming gjør det enkelt å distribuere og vedlikeholde applikasjonen, mens våre sikkerhetstiltak beskytter brukernes data mot potensielle trusler. Dette gjør Sparesti til en fremtidsrettet og pålitelig plattform for alle brukere.

Last edited by  [Emil Skogheim](#) 3 days ago

## Prosjektstruktur

Vår applikasjon følger en modulær og velorganisert arkitektur som sikrer skalerbarhet og vedlikeholdbarhet. Nedenfor er en oversikt over hovedkomponentene i både frontend og backend, noe som illustrerer en felles struktur som anvendes i våre prosjekter for å sikre konsistens og effektiv utvikling.

```
.
├── CONTRIBUTING.md
├── Frontend
│   ├── Dockerfile
│   ├── README.md
│   ├── coverage
│   │   ├── base.css
│   │   ├── block-navigation.js
│   │   ├── clover.xml
│   │   ├── coverage-final.json
│   │   ├── favicon.png
│   │   ├── index.html
│   │   ├── prettify.css
│   │   ├── prettify.js
│   │   ├── sort-arrow-sprite.png
│   │   └── sorter.js
│   └── src
│       ├── App.vue.html
│       └── components
│           ├── AddFriends
│           │   ├── FollowButton.vue.html
│           │   ├── ProfileCard.vue.html
│           │   ├── SearchBar.vue.html
│           │   ├── ShowFriends.vue.html
│           │   └── index.html
│           ├── FriendRequest
│           │   ├── AcceptButton.vue.html
│           │   ├── FriendRequestCard.vue.html
│           │   └── index.html
│           ├── HelloWorld.vue.html
│           ├── MainComponents
│           │   ├── Footer.vue.html
│           │   ├── Navbar.vue.html
│           │   └── index.html
│           ├── Newsfeed
│           │   ├── NewsFeed.vue.html
│           │   └── index.html
│           ├── SignUp
│           │   ├── PasswordCriteriaComponent.vue.html
│           │   ├── SignupHeaderComponent.vue.html
│           │   └── index.html
│           ├── UserConfig
│           │   ├── AnswerOption.vue.html
│           │   ├── ProgressBar.vue.html
│           │   └── index.html
│           └── separator.vue.html
│           └── badges
│               ├── AllBadges.vue.html
│               ├── BadgeProgressbar.vue.html
│               ├── YourBadges.vue.html
│               └── index.html
│           └── challenges
│               ├── AIChallenge.vue.html
│               ├── AddChallengesModal.vue.html
│               └── AddCustomChallenge.vue.html
└── W22
```

```
    └── FeedItem.vue.html
    └── GoalComponent.vue.html
    └── PostComponent.vue.html
    └── PostDisplay.vue.html
    └── addPostComponent.vue.html
    └── index.html

    └── goals
        └── AddGoals.vue.html
        └── GoalsDisplayer.vue.html
        └── SuccessCelebration.vue.html
        └── completedGoalModal.vue.html
        └── index.html
    └── index.html

    └── input
        └── InputFieldComponent.vue.html
        └── index.html
    └── landingPageComponents
        └── GoalsShowbox.vue.html
        └── InspirationGoals.vue.html
        └── ShowGoalsDisplayer.vue.html
        └── WelcomeIntro.vue.html
        └── index.html
    └── profile
        └── index.html
        └── updateProfileModal.vue.html
    └── streaks
        └── Streaks.vue.html
        └── index.html
    └── util
        └── StarRating.vue.html
        └── index.html
    └── index.html
    └── main.js.html
    └── router
        └── index.html
        └── index.js.html
    └── services
        └── api
            └── azureService.js.html
            └── index.html
            └── newsService.js.html
            └── openAIService.js.html
        └── baseService.js.html
        └── challenge
            └── challengeService.js.html
            └── dailyChallengeService.js.html
            └── index.html
            └── userChallengeService.js.html
        └── community
            └── GoalService.js.html
            └── PostService.js.html
            └── fetchFeedbackServices.js.html
            └── fetchUsersServices.js.html
            └── friendshipService.js.html
            └── index.html
        └── economicHabit
            └── economicHabitService.js.html
            └── index.html
            └── userEconomicHabitService.js.html
        └── friendshipService.js.html
        └── goal
            └── completeGoalService.js.html
            └── goalService.js.html
            └── index.html
            └── sharedGoalService.js.html
```

```
    └── badgeService.js.html
    └── index.html
    └── totalMoneySavedService.js.html
    └── userConfigService.js.html
    └── userService.js.html
    └── utils
        └── fileService.js.html
        └── index.html
    └── stores
        └── auth.js.html
        └── badge.js.html
        └── challengeStore.js.html
        └── dailyChallengeStore.js.html
        └── economicHabit.js.html
        └── friendshipStore.js.html
        └── goalStore.js.html
        └── index.html
        └── totalMoneySavedStore.js.html
        └── userConfig.js.html
        └── userStore.js.html
    └── utils
        └── dateUtils.js.html
        └── index.html
    └── views
        └── AboutUsView
            └── AboutUsView.vue.html
            └── index.html
        └── FeedbackView
            └── FeedbackView.vue.html
            └── index.html
        └── ForgotPasswordView
            └── ForgotPasswordView.vue.html
            └── index.html
        └── FriendRequest
            └── FriendRequestView.vue.html
            └── index.html
        └── NewsFeed
            └── NewsFeedView.vue.html
            └── index.html
        └── TwoFAView
            └── TwoFAView.vue.html
            └── index.html
        └── UserConfig
            └── UserConfig.vue.html
            └── index.html
        └── addFriendsView
            └── AddFriendsView.vue.html
            └── index.html
        └── community
            └── SocialPage.vue.html
            └── addPostView.vue.html
            └── index.html
        └── friendsView
            └── FriendsView.vue.html
            └── index.html
        └── gameView
            └── GameView.vue.html
            └── ProgressBar.vue.html
            └── ProgressPath.vue.html
            └── TaskList.vue.html
            └── index.html
        └── goalView
            └── GoalView.vue.html
            └── index.html
        └── homeView
```

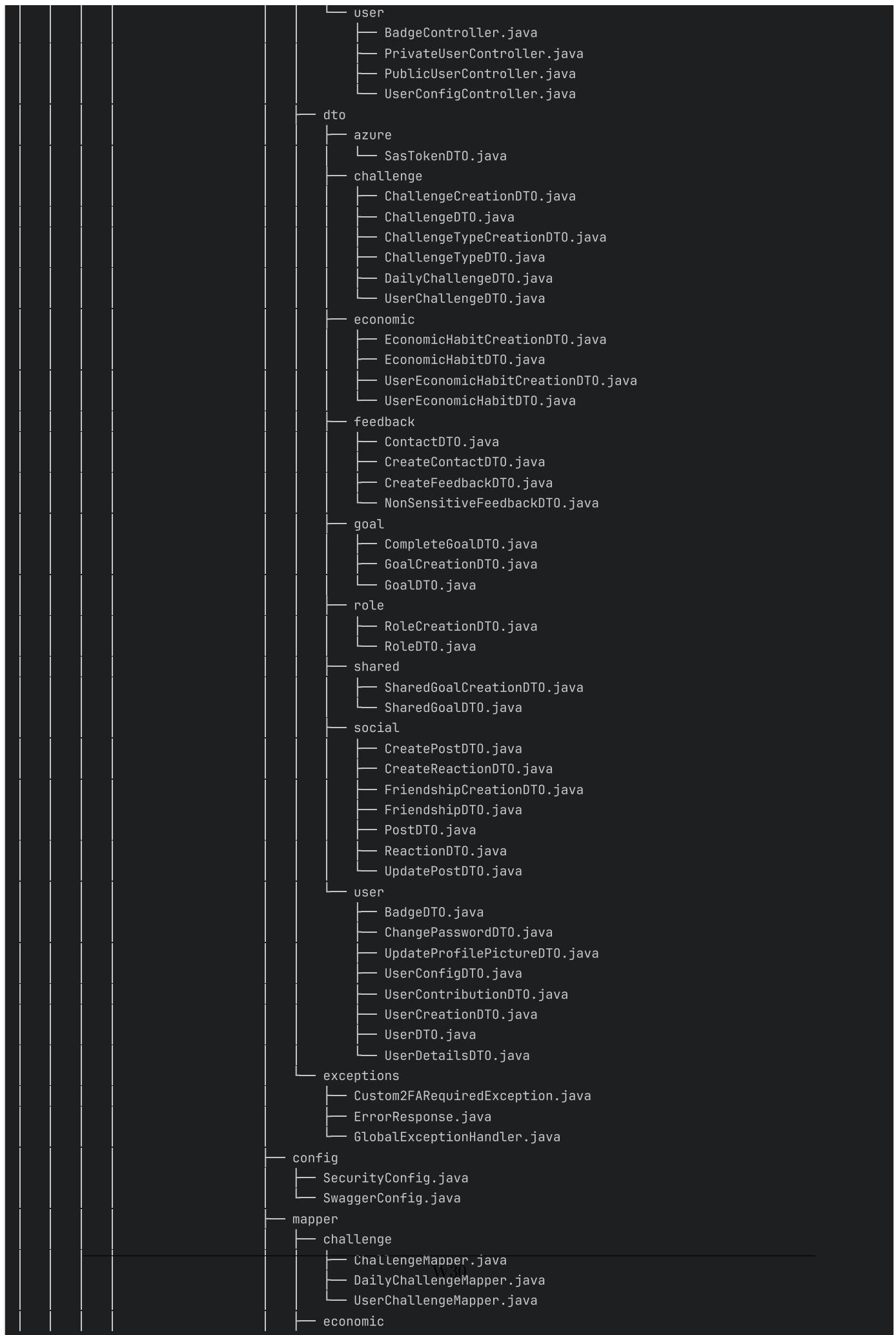
```
        └── landingPage
            ├── LandingPage.vue.html
            └── index.html
        └── loginView
            ├── LoginPageView.vue.html
            └── index.html
        └── profilePage
            ├── ProfilePage.vue.html
            └── index.html
        └── signupView
            ├── SignUpView.vue.html
            └── index.html
    └── cypress
        ├── e2e
            ├── 2FA.cy.js
            └── signUp.cy.js
        ├── fixtures
            └── example.json
        ├── integration
            ├── login.spec.js
            ├── register.spec.js
            └── twofactor.spec.js
        └── support
            ├── commands.js
            └── e2e.js
    └── cypress.config.js
    └── default.conf
    └── index.html
    └── package-lock.json
    └── package.json
    └── public
        ├── JSON
            ├── FriendRequestData.json
            ├── TestProfiles.json
            ├── allbadges.json
            ├── challenges.json
            ├── popularChallenges.json
            ├── questions.json
            └── yourChallenges.json
        └── images
            ├── LandingPageImages
                └── PigOnPath.jpeg
            ├── badges
                ├── first_goal.png
                ├── save1000.png
                └── save5000.png
            ├── coin.png
            ├── gate.png
            ├── gif
                └── ai-generation.gif
            ├── hawaii.jpg
            ├── img.png
            ├── logo
                ├── cleaner_logo.png
                ├── cleaner_logo_notext.png
                ├── cleaner_logo_notext_bluish.png
                └── logo.png
            ├── palm-tree-vector-icon.jpg
            ├── path.png
            ├── pexels-photo-1940041.jpeg
            ├── pexels-pixabay-35967.jpg
            ├── pig.png
            ├── piggybank.webp
            ├── porche.jpg
            └── profile-pic.jpg
```

```
team_photos
  elias.jpeg
  emil.jpeg
  erik.jpeg
  jacob.jpeg
  kristoffer.jpeg
  sander.jpeg
  talian.jpeg
  vegard.jpeg
sounds
  pig-sound.mp3
vite.svg
src
  App.vue
  assets
    userProfile.png
    vue.svg
  components
    AddFriends
      FollowButton.vue
      ProfileCard.vue
      SearchBar.vue
      ShowFriends.vue
    FriendRequest
      AcceptButton.vue
      FriendRequestCard.vue
    HelloWorld.vue
    MainComponents
      Footer.vue
      Navbar.vue
    Newsfeed
      NewsFeed.vue
    SignUp
      PasswordCriteriaComponent.vue
      SignupHeaderComponent.vue
    UserConfig
      AnswerOption.vue
      ProgressBar.vue
      separator.vue
    badges
      AllBadges.vue
      BadgeProgressbar.vue
      YourBadges.vue
    challenges
      AIChallenge.vue
      AddChallengesModal.vue
      AddCustomChallenge.vue
    community
      FeedItem.vue
      GoalComponent.vue
      PostComponent.vue
      PostDisplay.vue
      addPostComponent.vue
    goals
      AddGoals.vue
      GoalsDisplayer.vue
      SuccessCelebration.vue
      completedGoalModal.vue
    input
      InputFieldComponent.vue
    landingPageComponents
      GoalsShowbox.vue
      InspirationGoals.vue
      ShowGoalsDisplayer.vue
      WelcomeIntro.vue
```

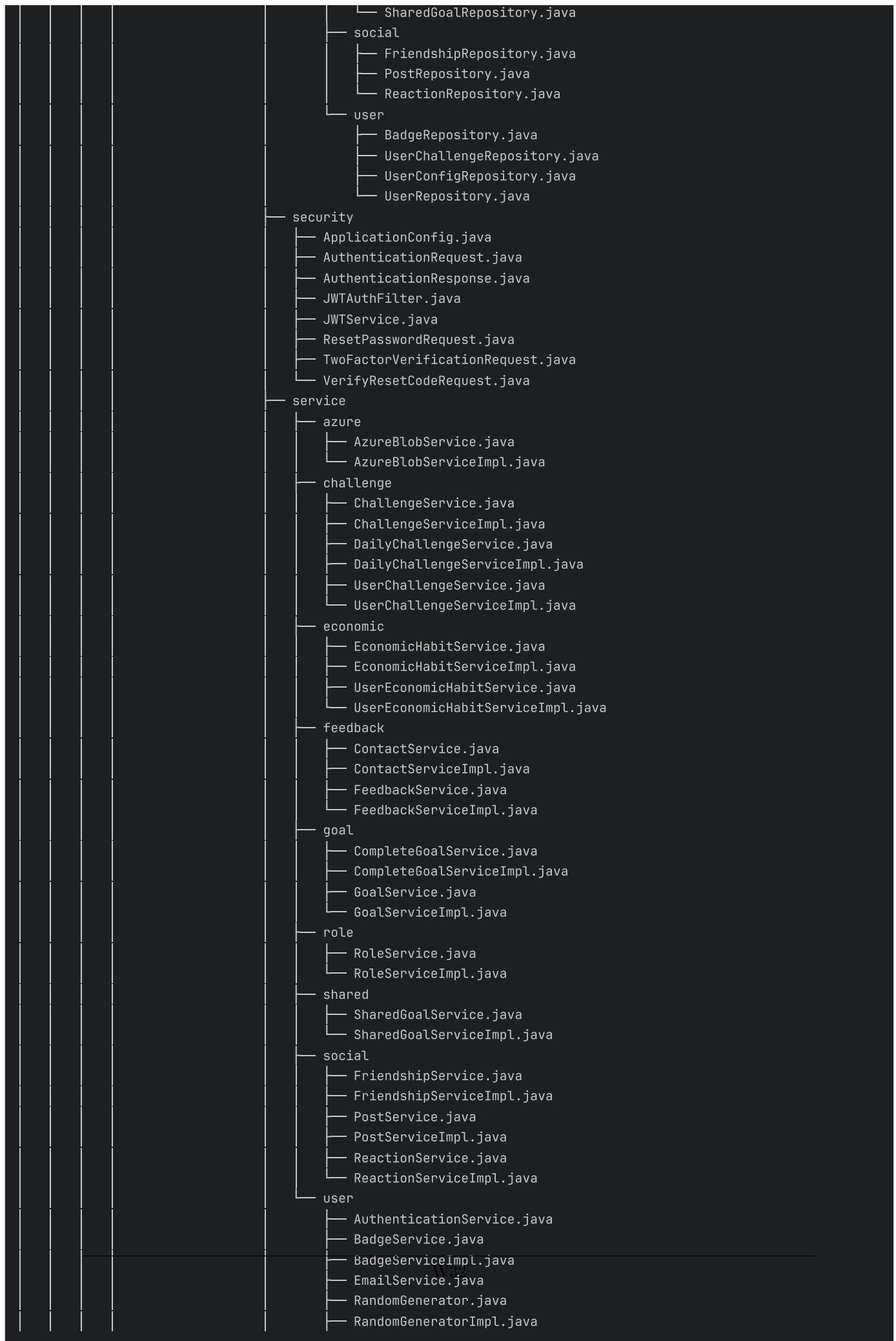
```
streaks
  └── Streaks.vue
util
  └── StarRating.vue
main.js
router
  └── index.js
services
  ├── api
  │   ├── azureService.js
  │   ├── newsService.js
  │   └── openAIService.js
  ├── baseService.js
  ├── challenge
  │   ├── challengeService.js
  │   ├── dailyChallengeService.js
  │   └── userChallengeService.js
  ├── community
  │   ├── GoalService.js
  │   ├── PostService.js
  │   ├── fetchFeedbackServices.js
  │   ├── fetchUsersServices.js
  │   └── friendshipService.js
  ├── economicHabit
  │   ├── economicHabitService.js
  │   └── userEconomicHabitService.js
  ├── friendshipService.js
  ├── goal
  │   ├── completeGoalService.js
  │   ├── goalService.js
  │   └── sharedGoalService.js
  ├── user
  │   ├── badgeService.js
  │   ├── totalMoneySavedService.js
  │   ├── userConfigService.js
  │   └── userService.js
  └── utils
      └── fileService.js
stores
  ├── auth.js
  ├── badge.js
  ├── challengeStore.js
  ├── dailyChallengeStore.js
  ├── economicHabit.js
  ├── friendshipStore.js
  ├── goalStore.js
  ├── totalMoneySavedStore.js
  ├── userConfig.js
  └── userStore.js
style.css
utils
  └── dateUtils.js
views
  ├── AboutUsView
  │   └── AboutUsView.vue
  ├── FeedbackView
  │   └── FeedbackView.vue
  ├── ForgotPasswordView
  │   └── ForgotPasswordView.vue
  ├── FriendRequest
  │   └── FriendRequestView.vue
  ├── NewsFeed
  │   └── NewsFeedView.vue
  └── TwoFAView
      └── TwoFAView.vue
```

```
    └── addFriendsView
        └── AddFriendsView.vue
    └── community
        └── SocialPage.vue
    └── gameView
        ├── GameView.vue
        ├── ProgressBar.vue
        ├── ProgressPath.vue
        └── TaskList.vue
    └── goalView
        └── GoalView.vue
    └── homeView
        └── Home.vue
    └── landingPage
        └── LandingPage.vue
    └── loginView
        └── LoginPageView.vue
    └── profilePage
        └── ProfilePage.vue
    └── signupView
        └── SignUpView.vue
└── test
    ├── components
        ├── AddFriends
            ├── FollowButton.test.js
            ├── SearchBar.test.js
            └── ShowFriends.test.js
        ├── FriendRequest
            ├── AcceptButton.test.js
            └── FriendsRequestCard.test.js
        ├── MainComponents
            └── Footer.test.js
        ├── NewsFeed
            └── NewsFeed.test.js
        ├── SignUp
            ├── PasswordCriteriaComponent.test.js
            └── SignupHeaderComponent.test.js
        ├── UserConfig
            ├── AnswerOption.test.js
            └── ProgressBar.test.js
        ├── badges
            └── AllBadges.test.js
        ├── challenges
            └── AddCustomChallenge.test.js
        ├── community
            ├── FeedItem.test.js
            ├── GoalComponent.test.js
            ├── PostComponent.test.js
            └── addPostComponent.test.js
        ├── goals
        ├── input
            └── InputFieldComponent.test.js
        ├── profile
            └── updateProfileModal.test.js
        ├── streaks
            └── Streaks.test.js
        └── util
            └── StarRating.test.js
    └── services
        ├── api
            ├── azureService.test.js
            └── newsService.test.js
        └── goal
            ├── completeGoalService.test.js
            └── goalService.test.js
```

```
    └── userService.test.js
  └── stores
    ├── auth.test.js
    ├── challengeStore.test.js
    ├── dailyChallengeStore.test.js
    ├── friendshipStore.test.js
    ├── goalStore.test.js
    └── totalMoneySavedStore.test.js
  └── userStore.test.js
  └── views
    ├── AboutUsView
    │   └── AboutUsView.test.js
    ├── FeedbackView
    │   └── FeedbackView.test.js
    ├── ForgotPasswordView
    │   └── ForgotPasswordView.test.js
    ├── GoalView
    │   └── GoalView.test.js
    ├── LandingPage
    │   └── LandingPage.test.js
    ├── SocialPage
    │   └── socialPageView.test.js
    ├── addFriendsView
    │   └── addFriendsView.test.js
    └── gameView
        └── ProgressBar.test.js
  └── vite.config.js
  └── Makefile
  └── README.md
  └── backend
    ├── Dockerfile
    ├── mvnw
    ├── mvnw.cmd
    ├── pom.xml
    └── src
      └── main
        └── java
          └── edu
            └── ntnu
              └── idatt2106_2024_05
                └── backend
                  ├── BackendApplication.java
                  └── api
                    ├── controller
                    │   ├── azure
                    │   │   └── StorageController.java
                    │   ├── challenge
                    │   │   ├── ChallengeController.java
                    │   │   ├── DailyChallengeController.java
                    │   │   └── UserChallengeController.java
                    │   ├── economic
                    │   │   ├── EconomicHabitController.java
                    │   │   └── UserEconomicHabitController.java
                    │   ├── feedback
                    │   │   ├── ContactController.java
                    │   │   └── FeedbackController.java
                    │   ├── goal
                    │   │   ├── CompleteGoalController.java
                    │   │   └── GoalController.java
                    │   ├── role
                    │   │   └── RoleController.java
                    │   └── shared
                    │       └── SharedGoalController.java
                    └── social
                        └── FriendshipController.java
```





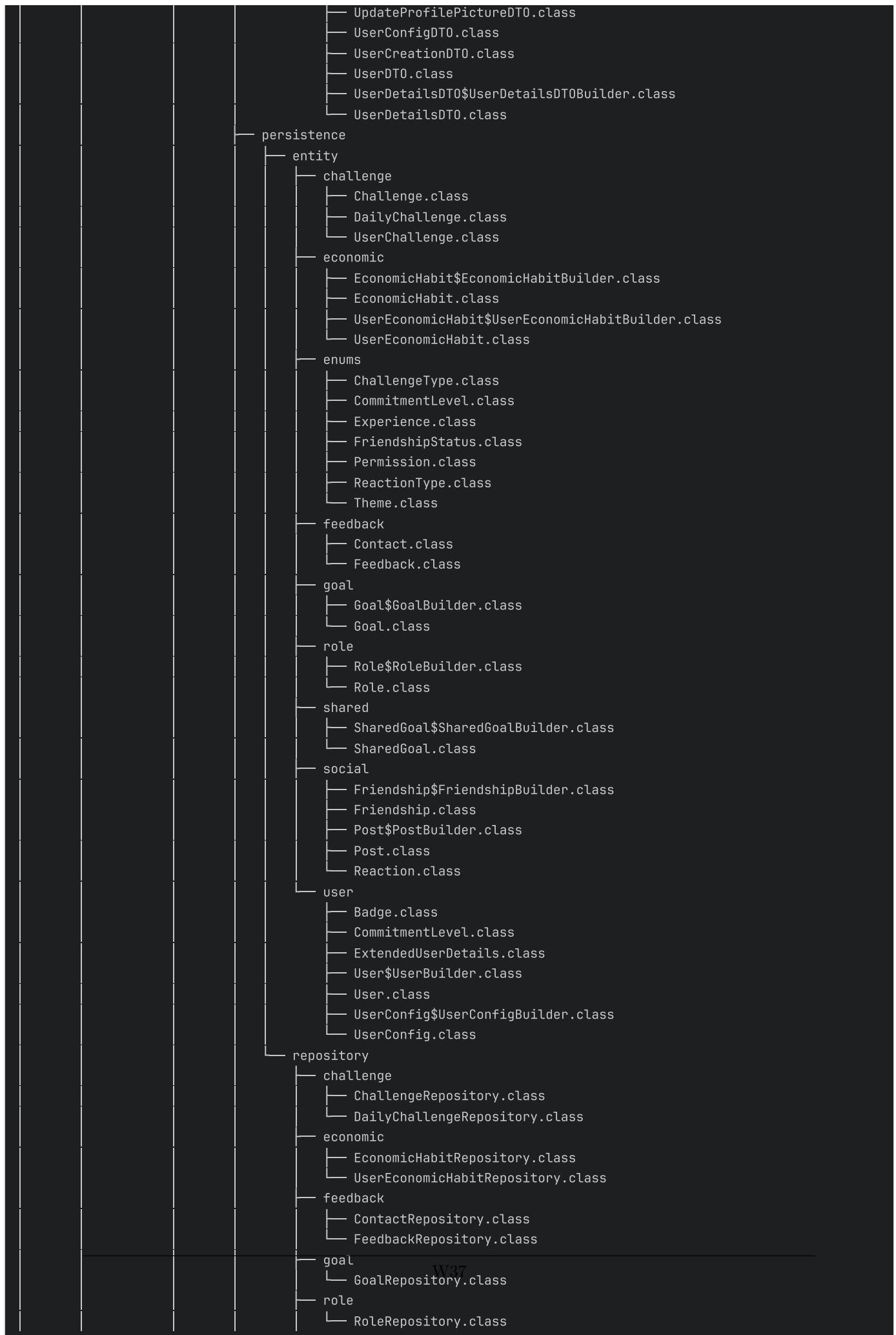


















```
└── service
    ├── challenge
    │   ├── ChallengeServiceImplTest.class
    │   ├── DailyChallengeServiceImplTest.class
    │   └── UserChallengeServiceImplTest.class
    ├── economic
    │   ├── EconomicHabitServiceImplTest.class
    │   └── UserEconomicHabitServiceImplTest.class
    ├── feedback
    ├── goal
    │   ├── CompleteGoalServiceImplTest.class
    │   └── GoalServiceImplTest.class
    ├── shared
    │   └── SharedGoalServiceImplTest.class
    ├── social
    │   ├── FriendshipServiceImplTest.class
    │   ├── PostServiceImplTest.class
    │   └── ReactionServiceImplTest.class
    └── user
        ├── AuthenticationServiceTest.class
        ├── BadgeServiceImplTest.class
        ├── UserConfigServiceImplTest.class
        └── UserServiceImplTest.class

```

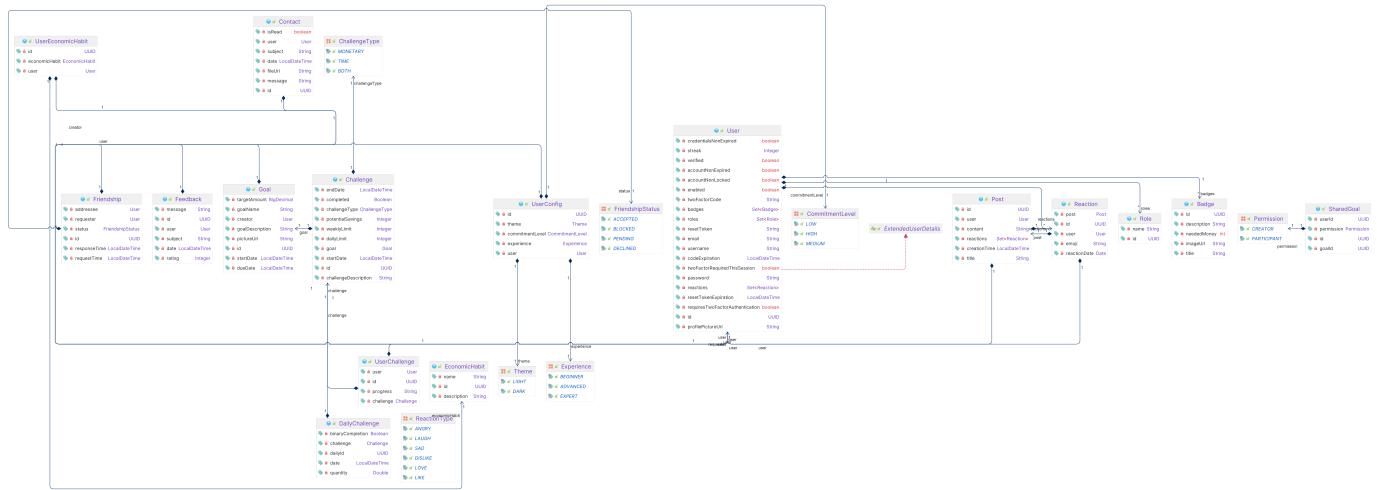
cleaner\_logo.png  
docker-compose.yml  
tree.md

392 directories, 897 files

Last edited by  [Sander Rom Skofsrud](#) 1 day ago

# Klassediagram

Under vises alle klassene i models mappen. Disse klassene representerer hver en entitet i databasen.



---

Last edited by  Jacob Forsdahl Iqbal 26 minutes ago

# Servertjenester

Sparesti er bygget på en solid infrastruktur. Den inkluderer flere servertjenester for å sikre høy tilgjengelighet, ytelse og sikkerhet. Denne siden gir en detaljert oversikt over de ulike servertjenestene som er implementert.

## Oversikt over servertjenester

- **Web Server**
  - Nginx
- **Application Server**
  - Spring Boot
- **Database Server**
  - MySQL
- **Containerization**
  - Docker og Docker Compose
- **Cloud Storage**
  - Azure Blob Storage

## Web Server

- **Nginx**
  - Sparesti benytter Nginx som webserver for å håndtere HTTP-forespørsler. Nginx tilbyr høy ytelse og pålitelighet.
- **Lastbalansering**
  - Nginx tilbyr funksjonalitet for distribuerere trafikk jevnt over flere applikasjonsservere, noe som sikrer at ingen enkelt server blir overbelastet. Dette er ikke implementert eller testet, men ettersom nginx er valgt, så er dette svært lett å sette opp i et produksjonsmiljø.
- **Statisk innhold**
  - Har statiske filer som bilder, CSS og JavaScript direkte fra webserveren for raskere lastetider.
- **SSL/TLS**
  - Håndterer HTTPS-forespørsler for å sikre sikker kommunikasjon mellom kunder og servere.

## Applikasjon Server

- **Spring Boot**
  - Sparesti bruker Spring Boot som applikasjonserver for å kjøre backend-tjenestene.
- **REST API**
  - Håndterer alle API-forespørsler fra frontenden og eksterne tjenester.
- **Forretningslogikk**
  - Inneholder all logikk for applikasjonens funksjoner, som brukerhåndtering, transaksjoner og meldingsutsending.
- **Autentisering og Autorisasjon**
  - Bruker Spring Security for å administrere brukerautentisering, autorisasjon og to-faktor autentisering (2FA).

## Database Server

- **MySQL**
  - Sparesti benytter MySQL som database for å lagre all vedvarende data.
- **Datahåndtering**
  - Lagre og hente brukerdata, transaksjonslogg og annen relevant informasjon.
- **Ytelsesoptimalisering**
  - Indeksering og spørringsoptimalisering for å sikre rask datatilgang.

## Containerization

- **Docker**

---

  - Alle servertjenester i Sparesti kjører i Docker-containere for å sikre enkelt oppsett og god sikkerhet.
- **Isolasjon**
  - Hver tjeneste kjører i sin egen container, noe som sikrer at de er isolert fra hverandre og kan administreres individuelt.

- **Portabilitet**
  - Docker-containere kan kjøre på ulike plattformer, noe som gjør det enkelt å utvikle og kjøre applikasjonen på for eksempel en linux server.
- **Docker Compose**
  - Brukes til å definere og administrere applikasjoner bestående av flere containere, inkludert nettverk og volumer som kreves for å kjøre Sparesti.
  - **Tjenestedefinisjon**
    - Konfigurerer alle nødvendige tjenester (web, applikasjon, database) i en enkel YAML-fil.
  - **Avhengighetshåndtering**
    - Sikrer at tjenestene startes og stoppes i riktig rekkefølge, basert på avhengigheter.

## Cloud Storage

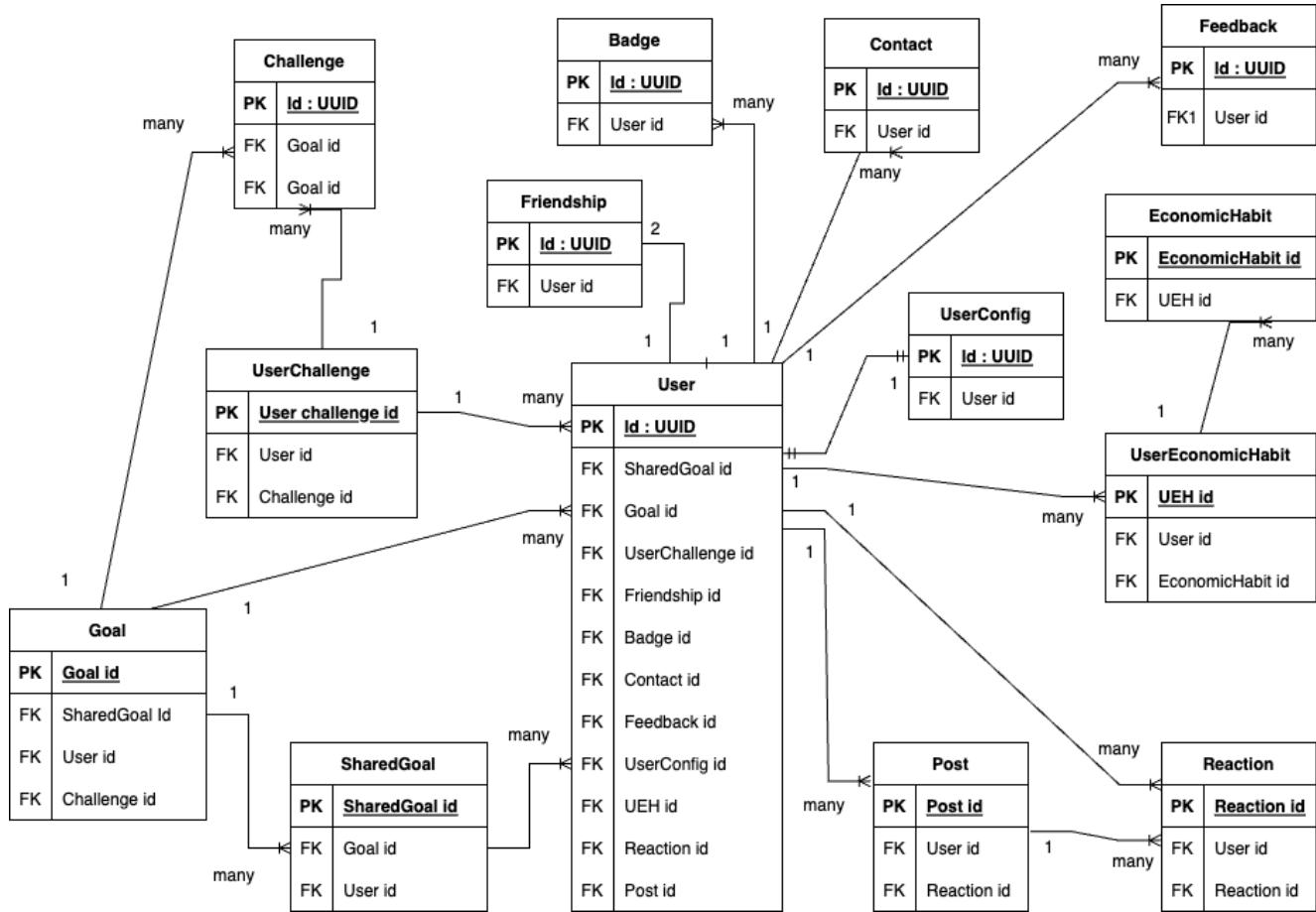
- **Azure Blob Storage**
  - Brukes for å lagre bilder. Azure Blob Storage gir en skalerbar og sikker løsning for lagring av ustrukturert data.
  - **Sikkerhet**
    - Bruk av SAS-tokens (Shared Access Signature) for å begrense tilgang til lagrede filer.
  - **Høy tilgjengelighet**
    - Sørger for at data alltid er tilgjengelig for applikasjonen.
  - **Skalerbarhet**
    - Tilbyr potensielt ubegrenset lagringskapasitet, slik at applikasjonen kan vokse uten å bekymre seg for lagringsbegrensninger.

## Oppsummert

Servertjenestene i Sparesti er nøye utvalgt og konfigurert for å sikre en pålitelig, sikker og skalerbar plattform. Ved å bruke en kombinasjon av Nginx, Spring Boot, MySQL, Docker og Azure Blob Storage gir det applikasjonen en solid infrastruktur som møter behovene til brukere og teamet.

Last edited by  [Vegard Johnsen](#) 7 hours ago

## Databasemodell



Last edited by  [Vegard Johnsen](#) 12 hours ago

# CI og testing

## Generelt

Denne CI/CD-pipelinen er konfigurert for automatisk bygging, testing, pakking og distribusjon av vår programvare. Pipelinen er aktiv på både dev og main grener, med spesifikke distribusjonsregler kun for main grenen.

## Stadier

### Build

- build-backend: Kompiler backend med Maven.
- build-frontend: Bygger frontend med Node.js og lagrer artifaktene.

### Test

- test-backend: Kjører backend-tester med Maven.
- test-frontend-unit: Kjører enhetstester for frontend med Vitest.
- test-frontend-e2e: Utfører ende-til-ende tester med Cypress.

### Package

- package-backend: Pakker backend-koden i en .jar fil uten å kjøre tester.

### Deploy

- deploy-job: Utfører distribusjon til produksjon kun fra main grenen.

## Regler

- Jobber kjører på både dev og main grener, men deploy skjer kun fra main.
- Artifakter fra bygge- og pakkesteg lagres og er tilgjengelige for nedlasting i opptil en uke.

## Testing og Kodedekning

### Frontend

Rapporten under viser dekningsgraden av testene på frontend.

File	% Stmt	% Branch	% Funcs	% Lines	Uncovered	Line #s
All files	82.13	81.28	36.17	82.13		
src	0	0	0	0		
App.vue	0	0	0	0	1-34	
main.js	0	0	0	0	1-26	
src/components	0	0	0	0		
HelloWorld.vue	0	0	0	0	1-41	
src/components/AddFriends	97.12	91.66	50	97.12		
FollowButton.vue	100	100	100	100		
ProfileCard.vue	100	100	100	100		
SearchBar.vue	96.09	100	25	96.09	40-41,48-49,60	
ShowFriends.vue	93.95	87.5	66.66	93.95	38-39,66-72	
src/components/FriendRequest	92.82	50	33.33	92.82		
AcceptButton.vue	100	100	100	100		
FriendRequestCard.vue	90	0	0	90	47-53,59-65	
src/components/MainComponents	23.22	66.66	0	23.22		
Footer.vue	96.72	100	0	96.72	34-35,41-42	
Navbar.vue	0	0	0	0	1-386	
src/components/Newsfeed	98.96	80	50	98.96		
NewsFeed.vue	98.96	80	50	98.96	58-59	
src/components/SignUp	100	100	100	100		
PasswordCriteriaComponent.vue	100	100	100	100		
SignupHeaderComponent.vue	100	100	100	100		
src/components/UserConfig	90.37	66.66	50	90.37		
AnswerOption.vue	100	100	100	100		

src/components/badges	99.64	83.33	100	99.64	
AllBadges.vue	99.2	80	100	99.2	1
BadgeProgressbar.vue	100	100	100	100	
YourBadges.vue	100	100	100	100	
src/components/challenges	76.19	85.1	28.57	76.19	
AIChallenge.vue	0	0	0	0	1-9
AddChallengesModal.vue	73.2	90	13.33	73.2	...136,143-152,159-166,173-175,180
AddCustomChallenge.vue	86.13	86.11	50	86.13	1-26,103-107,140-142,145-147,153
src/components/community	92.41	71.87	88.88	92.41	
FeedItem.vue	97.36	50	100	97.36	1
GoalComponent.vue	86.63	55.55	100	86.63	48-54,56-57,64-72,81-87
PostComponent.vue	97.2	63.63	100	97.2	63-64,79-80,82-83
PostDisplay.vue	100	100	100	100	
addPostComponent.vue	89.95	100	80	89.95	54-76
src/components/goals	91.45	87.5	4.16	91.45	
AddGoals.vue	74.77	83.33	4.16	74.77	...230,235-260,265-276,281-284,289
GoalsDisplayer.vue	100	100	100	100	
SuccessCelebration.vue	100	100	100	100	
completedGoalModal.vue	100	100	100	100	
src/components/input	100	100	100	100	
InputFieldComponent.vue	100	100	100	100	
src/components/LandingPageComponents	17.46	25	25	17.46	
GoalsShowbox.vue	0	0	0	0	1-130
InspirationGoals.vue	0	0	0	0	1-53
ShowGoalsDisplayer.vue	0	0	0	0	1-181
WelcomeIntro.vue	100	100	100	100	
src/components/profile	98.8	89.65	100	98.8	
updateProfileModal.vue	98.8	89.65	100	98.8	47,121,142
src/components/streaks	100	100	100	100	
Streaks.vue	100	100	100	100	
src/components/util	100	100	100	100	
StarRating.vue	100	100	100	100	
src/router	86	100	100	86	
index.js	86	100	100	86	123-143
src/services	67.05	63.63	30.76	67.05	
baseService.js	61.32	63.63	50	61.32	26-30,38-43,54,61-68,71-77,80-86,89
friendshipService.js	76.11	100	0	76.11	21-27,35,45-49,57,65-66
src/services/api	73.17	50	66.66	73.17	
azureService.js	95.58	50	100	95.58	22,46,62
newsService.js	90	50	100	90	24-26
openAIService.js	54.2	100	0	54.2	51-82,90-106
src/services/challenge	85.04	100	27.27	85.04	
challengeService.js	82.6	100	20	82.6	17-18,27-28,35-36,42-43
dailyChallengeService.js	85	100	25	85	17-18,27-28,36-37
userChallengeService.js	90.47	100	50	90.47	17-18
src/services/community	79.72	100	20	79.72	
GoalService.js	81.08	100	12.5	81.08	17-18,26-27,36-37,45-46,53-54,62-63
PostService.js	80.51	100	12.5	80.51	17-18,25-26,33-34,43-44,52-53,62-63
fetchFeedbackServices.js	86.66	100	33.33	86.66	17-18,26-27
fetchUsersServices.js	86.66	100	33.33	86.66	17-18,26-27
friendshipService.js	72.94	100	25	72.94	18-20,27-28,37-40,47-48,64-73,81-82
src/services/economicHabit	91.11	100	50	91.11	
economicHabitService.js	90.9	100	50	90.9	16-17
userEconomicHabitService.js	91.3	100	50	91.3	17-18
src/services/goal	91.83	100	60	91.83	
completeGoalService.js	100	100	100	100	
goalService.js	89.47	100	50	89.47	35-36,43-44,51-52
sharedGoalService.js	90.47	100	50	90.47	17-18
src/services/user	74.88	88.23	37.83	74.88	
badgeService.js	100	100	100	100	
totalMoneySavedService.js	84.61	100	20	84.61	18-19,27-28,38-39,48-49
userConfigService.js	91.3	100	50	91.3	17-18
userService.js	66.78	77.77	25	66.78	...125,133-134,142-143,152-164,171
src/services/utils	24.07	100	0	24.07	
fileService.js	24.07	100	0	24.07	13-53

badge.js	63.46	100	0	63.46	9-10,19-23,32-36,45-51,61-70,79-84
challengeStore.js	96.77	100	66.66	96.77	28
dailyChallengeStore.js	59.06	100	18.18	59.06	12-26,34-39,69,79-80,89-90,99,108
economicHabit.js	64.15	100	33.33	64.15	24-32,40-49
friendshipStore.js	75	66.66	66.66	75	22-23,36-45
goalStore.js	70.28	100	30	70.28	24-37,45-53,62-67,76-79,102,111,112
totalMoneySavedStore.js	94.59	75	100	94.59	23-24
userConfig.js	64.15	100	33.33	64.15	22-30,38-47
userStore.js	93.1	100	80	93.1	61-66
src/utils	62.29	100	0	62.29	
dateUtils.js	62.29	100	0	62.29	15-21,31-32,42-44,51-61
src/views/AboutUsView	97.65	87.5	0	97.65	
AboutUsView.vue	97.65	87.5	0	97.65	81-82,105-107
src/views/FeedbackView	78.82	100	0	78.82	
FeedbackView.vue	78.82	100	0	78.82	1-16,53-59,65-85,91-93
src/views/ForgotPasswordView	78.61	76.74	59.09	78.61	
ForgotPasswordView.vue	78.61	76.74	59.09	78.61	1-48,120-121,124-127,183-185,206-207
src/views/FriendRequest	100	100	100	100	
FriendRequestView.vue	100	100	100	100	
src/views/NewsFeed	100	100	100	100	
NewsFeedView.vue	100	100	100	100	
src/views/TwoFAView	100	100	100	100	
TwoFAView.vue	100	100	100	100	
src/views/UserConfig	100	100	100	100	
UserConfig.vue	100	100	100	100	
src/views/addFriendsView	24.39	41.66	50	24.39	
AddFriendsView.vue	24.39	41.66	50	24.39	1-93
src/views/community	71.24	84.21	50	71.24	
SocialPage.vue	71.24	84.21	50	71.24	1-31,65-66,77-78,92-139,153-154,160
src/views/gameView	100	100	100	100	
GameView.vue	100	100	100	100	
ProgressBar.vue	100	100	100	100	
ProgressPath.vue	100	100	100	100	
TaskList.vue	100	100	100	100	
src/views/goalView	95.71	83.33	20	95.71	
GoalView.vue	95.71	83.33	20	95.71	43-44,50-51,60-61
src/views/homeView	100	100	100	100	
Home.vue	100	100	100	100	
src/views/landingPage	99.32	100	50	99.32	
LandingPage.vue	99.32	100	50	99.32	10-11
src/views/loginView	100	100	100	100	
LoginPageView.vue	100	100	100	100	
src/views/profilePage	100	100	100	100	
ProfilePage.vue	100	100	100	100	
src/views/signupView	100	100	100	100	
SignUpView.vue	100	100	100	100	

## Backend

For å måle dekningsgraden til backend-testene våre brukte vi både IntelliJ og Jacoco sin testrapport. Disse viser imidlertid noe forskjellige dekningsgrad, der IntelliJ viser 81% og Jacoco viser 50%. Dette er antageligvis på grunn av at rapportene behandler de autogenererte metodene lombok generer på forskjellig måte.

all	97% (144/147)	81% (714/876)	62% (1633/2616)
edu	97% (144/147)	81% (714/876)	62% (1633/2616)
ntnu	97% (144/147)	81% (714/876)	62% (1633/2616)
idatt2106_2024_05	97% (144/147)	81% (714/876)	62% (1633/2616)
backend	97% (144/147)	81% (714/876)	62% (1633/2616)
api	100% (63/63)	83% (326/392)	62% (470/756)
config	100% (3/3)	100% (8/8)	100% (41/41)
mapper	100% (16/16)	88% (64/72)	73% (444/603)
persistence	100% (32/32)	93% (181/193)	82% (231/281)
security	70% (7/10)	56% (28/50)	40% (39/96)
service	100% (21/21)	66% (103/156)	47% (388/818)
startup	100% (1/1)	100% (4/4)	100% (19/19)
BackendApplication	100% (1/1)	0% (0/1)	50% (1/2)

## Jacoco

## backend

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
edu.ntnu.idatt2106_2024_05.backend.api.dto.user	33 %	5 %	112	188	0	49	23	99	0	9	0	9
edu.ntnu.idatt2106_2024_05.backend.api.dto.social	29 %	0 %	92	152	0	43	17	77	0	7	0	7
edu.ntnu.idatt2106_2024_05.backend.api.dto.challenge	37 %	0 %	88	168	0	52	16	96	0	6	0	6
edu.ntnu.idatt2106_2024_05.backend.security	24 %	12 %	74	107	57	97	38	71	3	10	3	10
edu.ntnu.idatt2106_2024_05.backend.api.dto.feedback	25 %	0 %	74	110	2	24	21	57	0	5	0	5
edu.ntnu.idatt2106_2024_05.backend.service.goal	35 %	25 %	46	61	110	166	12	24	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.service.challenge	32 %	22 %	46	69	85	123	18	35	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.service.user	62 %	51 %	46	92	92	240	18	56	0	6	0	6
edu.ntnu.idatt2106_2024_05.backend.api.controller.challenge	10 %	n/a	19	25	83	89	19	25	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.api.dto.goal	37 %	0 %	45	95	1	32	6	56	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.user	72 %	34 %	37	123	11	79	24	107	0	5	0	5
edu.ntnu.idatt2106_2024_05.backend.api.controller.social	41 %	n/a	11	22	48	78	11	22	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.api.controller.economic	11 %	n/a	12	16	47	51	12	16	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.shared	28 %	0 %	26	41	0	8	7	22	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.service.shared	47 %	40 %	11	25	31	60	0	9	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.api.dto.shared	30 %	0 %	23	39	0	12	4	20	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.service.azure	1 %	0 %	6	7	44	45	4	5	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.api.controller.goal	22 %	n/a	8	13	39	49	8	13	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.api.controller.user	74 %	50 %	15	43	34	140	4	32	0	4	0	4
edu.ntnu.idatt2106_2024_05.backend.api.exceptions	65 %	0 %	22	42	1	53	7	27	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.mapper.social	65 %	39 %	37	53	45	131	3	19	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.feedback	37 %	0 %	19	40	19	33	11	32	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.mapper.challenge	77 %	51 %	34	58	38	161	0	21	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.service.economic	65 %	56 %	12	34	24	70	5	19	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.goal	58 %	6 %	12	41	14	27	4	33	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.service.feedback	16 %	8 %	12	15	25	28	6	9	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.social	79 %	61 %	15	77	5	48	6	64	0	5	0	5
edu.ntnu.idatt2106_2024_05.backend.service.social	80 %	65 %	11	44	21	108	2	25	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.api.dto.azure	18 %	0 %	15	20	0	4	4	9	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.mapper.shared	46 %	31 %	10	15	24	43	3	7	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.api.controller.feedback	27 %	n/a	4	8	16	20	4	8	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.api.controller.shared	12 %	n/a	4	5	15	16	4	5	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.mapper.feedback	77 %	43 %	15	25	17	66	1	9	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.service.role	11 %	0 %	6	7	11	12	2	3	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.mapper.economic	81 %	53 %	15	28	18	70	1	12	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.economic	77 %	n/a	5	30	0	14	5	30	0	4	0	4
edu.ntnu.idatt2106_2024_05.backend.mapper.user	89 %	57 %	17	32	11	78	0	12	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.api.dto.economic	91 %	n/a	2	38	0	20	2	38	0	6	0	6
edu.ntnu.idatt2106_2024_05.backend.mapper.goal	90 %	50 %	8	14	6	41	0	5	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.role	87 %	n/a	1	12	0	6	1	12	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.challenge	96 %	50 %	1	43	1	29	0	42	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend	37 %	n/a	1	2	2	3	1	2	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.config	100 %	n/a	0	9	0	41	0	9	0	3	0	3
edu.ntnu.idatt2106_2024_05.backend.persistence.entity.enums	100 %	n/a	0	7	0	30	0	7	0	7	0	7
edu.ntnu.idatt2106_2024_05.backend.startup	100 %	50 %	2	6	0	22	0	4	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.api.dto.role	100 %	n/a	0	10	0	9	0	10	0	2	0	2
edu.ntnu.idatt2106_2024_05.backend.api.controller.role	100 %	n/a	0	3	0	8	0	3	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.mapper.role	100 %	83 %	1	6	0	13	0	3	0	1	0	1
edu.ntnu.idatt2106_2024_05.backend.api.controller.azure	100 %	n/a	0	5	0	5	0	3	0	1	0	1

Total

8 452 of 17 176 50 % 1 273 of 1 598 20 % 1 072 2 123 997 2 646 334 1 324 3 147

---

Last edited by  [Lars Talian Stangebye-Hansen](#) 1 day ago

# Sikkerhet

I Sparesti-applikasjonen er det implementert flere sikkerhetstiltak for å sikre at brukerdata er beskyttet og at applikasjonen er robust mot angrep.

## Sikkerhetsarkitektur

**Backend:** Sparesti bruker Spring Boot som rammeverk for backenden, som kjører i Docker-containere for å isolere og skalerere tjenestene effektivt.

**Frontend:** Brukergrensesnittet er utviklet med Vue.js, som også kjøres i Docker-containere for å opprettholde konsistens og sikkerhet i produksjon.

**Database:** MySQL brukes for datalagring, hvor sensitiv informasjon som brukerdata håndteres i egen docker-konteiner for å unngå datalekasjer. Passord er kryptert og ID-er unike.

## Sikkerhetsfunksjoner

**Azure Blob Storage:** Bilder og andre mediefiler lagres trygt i Azure Blob Storage. Tilgang til disse filene er sikret med SAS-tokens (Shared Access Signature), som genererer tidsbegrensede og ressursspesifikke URL-er for å forhindre uautorisert tilgang.

**Autentisering og Autorisasjon:** Sparesti implementerer to-faktor autentisering (2FA) koblet til brukernes e-post for å forsterke sikkerheten. Dette sikkerhetstiltaket er styrbar fra backenden, og om en bruker trenger å gjennomføre 2FA ved neste innlogging er definert i et felt i brukerobjektet i databasen.

**OWASP Best Practices:** Applikasjonen følger best praksis anbefalt av Open Web Application Security Project (OWASP) for å forhindre vanlige sikkerhetsrisikoer.

**Private og Public User-Controllers:** Backend-arkitekturen benytter private og public controllers for å skille mellom autentiserte endepunkt. Alle endepunkt utenfor PublicController klassen krever autentisering med JWT token.

**JWT Tokens i Session Storage:** For å sikre at brukerens sesjonsdata håndteres sikkert, bruker Sparesti JSON Web Tokens (JWT) som lagres i session storage. JWT gir en sikker og kompakt metode for å overføre informasjon mellom frontend og backend. JWT-token har en begrenset levetid.

**Docker Containere:** Alle komponenter av Sparesti-applikasjonen kjører i isolerte Docker-containere, som bidrar til økt sikkerhet ved at hver del av applikasjonen kan kontrolleres og oppdateres uavhengig.

## Oppsummert

Sikkerhet er en grunnleggende del av Sparesti-applikasjonen, integrert i alle nivåer av utvikling og drift. Gjennom bruk av moderne teknologier og strenge sikkerhetsprotokoller, sikrer Sparesti at brukernes data og interaksjoner er beskyttet mot digitale trusler. Sparesti er dermed en svært sikker og robust applikasjon.