Prosjektbeskrivelse: "Self-Insight Journal" - En digital journal for åndelig vekst og selvinnsikt

Bakgrunn

Dette prosjektet skal lage en webapplikasjon som gir brukere et verktøy for å dokumentere og reflektere over sin åndelige reise og personlige vekst. Applikasjonen vil tilby funksjonalitet for journalføring, visualisering av fremgang og kobling til ulike verktøy for selvinnsikt som meditasjonsteknikker, affirmasjoner og mindfulness-øvelser.

Teknologier som skal brukes:

- **Frontend**: React (med hooks, context API, og React Router)
- **Backend**: **NestJS** (en moderne og kraftig backend-rammeverk for Node.js, med TypeScript) + **TypeORM** for databasehåndtering
- **Database**: PostgreSQL (Relasjonell database for lagring av journalinnhold og brukerdata)
- **Autentisering**: JWT-basert autentisering for sikre brukerøkter
- **Deploy**: Docker og Kubernetes for distribusjon og skalerbarhet
- **API**: RESTful API for kommunikasjon mellom frontend og backend
- **Verktøy**:
    - Redux (for global state management i React)
    - Chakra UI eller Material-UI for UI-komponenter
    - Docker for containerisering av applikasjonen

Funksjonalitet:

1. **Brukerautentisering og profilen**:
    - Brukerne skal kunne registrere seg, logge inn, og opprette en personlig profil.
    - Profilen skal inkludere informasjon om deres åndelige reise, og de skal kunne oppdatere dette gjennom sitt dashbord.
    - Autentisering gjøres med JWT for å sikre at brukerdata er trygt lagret og håndtert.
2. **Journalføring**:
    - Brukere kan skrive daglige refleksjoner om sine åndelige opplevelser og selvinnsikt.
    - De kan merke innholdet med tags som: meditasjon, affirmasjoner, visualisering, mindfulness, etc.
    - Brukeren skal kunne legge ved bilder, videoer eller lydfiler som en del av sine refleksjoner.
    - Søkefunksjonalitet for å finne journalinnlegg basert på tags, dato, eller spesifikke søkeord.

3. **Visualisering og fremgang**:
    - En grafisk fremstilling av brukerens refleksjoner og fremgang over tid, inkludert:
        - En tidslinje for når de skrev dagbokinnleggene sine
        - Analyse av hvilke tags eller temaer de skriver om mest
        - En progressbar som viser hvor ofte brukeren har logget innlegg (engasjement)
    - Et daglig "spiritual score" system som gir tilbakemelding på brukerens engasjement og innsikt.
4. **Meditasjon og affirmasjoner**:
    - Brukerne kan velge å få daglige påminnelser for meditasjonsøkter, mindfulness-øvelser, eller affirmationstekster.
    - Dette kan være en del av et større "søvn og mental helse"-modul som integrerer mindfulness-teknikker for bedre selvinnsikt.
5. **Interaktivt dashboard**:
    - Dashboardet skal vise brukeren deres daglige, ukentlige og månedlige fremgang.
    - Brukeren kan se hvilke områder de har utviklet seg i og hvilke områder som fortsatt trenger mer oppmerksomhet.
    - Brukeren kan sette personlige mål og få påminnelser om å reflektere regelmessig.
6. **Backend med NestJS og PostgreSQL**:
    - **NestJS** skal brukes til å bygge en robust REST API som håndterer alt fra brukerregistrering til journalinnlegg og fremgang.
    - **TypeORM** brukes til å interagere med PostgreSQL-databasen og administrere relasjonelle data som brukerinformasjon, journalinnlegg, tags, og meditasjon påminnelser.
    - API-et skal være godt strukturert, med klare "services", "controllers" og "modules".
7. **Testing og kvalitetssikring**:
    - Enhetstester (unit tests) for API-endepunktene, spesielt for autentisering, journalhåndtering og visualisering.
    - React-testing-library for å teste frontend-komponenter og deres interaktivitet.
    - Integrasjonstester for å sikre at frontend og backend fungerer sammen som forventet.
8. **Skalering og distribusjon**:
    - Bruk Docker for å containerisere både frontend- og backend-applikasjonen.
    - Bruk Kubernetes til å orkestrere distribusjon og skalering av applikasjonen, spesielt hvis applikasjonen skal vokse i antall brukere.
    - Deploy applikasjonen til en skyplattform som AWS, GCP eller Azure.

Teknisk utfordring:

Backend-rammeverket NestJS gir en avansert struktur for å bygge API-er, og kombinasjonen med TypeORM og PostgreSQL gir både skalerbarhet og muligheten for komplekse spørringer. Å bygge et RESTful API som håndterer relasjonelle data på en effektiv måte gir en fin utfordring i forhold til databasemodellering og autentisering.

På frontend-siden kan det være utfordrende å bygge et responsivt og brukervennlig dashboard i React, spesielt med fokus på visualisering av data som brukere kan bruke for å få innsikt i deres personlige vekst.

Bonusfunksjonalitet:

- **AI-drevet innsikt**: Bruk Maskinlæring (f.eks. TensorFlow.js) for å analysere dagbokinnleggene og gi brukeren tilbakemeldinger basert på mønstre og nøkkelord relatert til åndelighet og selvinnsikt.

Levering:

- En komplett applikasjon med frontend og backend fullt utviklet, inkludert autentisering, journalføring, og visualisering.
- Testrapporter og en grundig README-fil med instruksjoner for hvordan sette opp og kjøre applikasjonen lokalt.
- Deployert versjon av applikasjonen på en cloud-plattform (f.eks. AWS eller Heroku).