IT-Læring veiledning for Postly

1. Innledning

Formål med dokumentet

Dokumentet gir en veiledning for administrasjon av backend og serverinfrastruktur i prosjektet postly, Azure-hostede miljø, spesifikt basert på Debian 12-operativsystemet med tilgang via SSH fra Windows-terminalen. Målet er å støtte IT-personell, systemadministratorer og utviklere i å forstå og effektivt administrere de ulike komponentene i systemet.

Målgruppe og forutsetninger

Dette dokumentet er utviklet for IT-personell, systemadministratorer og utviklere som er ansvarlige for å administrere backend- og serverinfrastrukturen. Leserne forventes å ha grunnleggende kunnskap om webutvikling, Linux-operativsystemer, databaseadministrasjon og grunnleggende nettverkskonsepter. Videre forutsettes det at leserne er kjent med Azure-hostingmiljøer, Debian 12, SSH-tilkoblinger og har erfaring med PHP og MariaDB. Dette dokumentet gir en detaljert veiledning for å håndtere backend- og serveraspekter av systemet, og det antas at leserne har nødvendig kompetanse for å implementere de beskrevne løsningene.

Oversikt over backend og serverarkitektur

Her en oversikt over de sentrale komponentene og hva de gjør:

1. **Webserver:** Debian 12-operativsystemet, Apache, for å håndtere HTTP-forespørsler. Denne komponenten fungerer som grensesnittet mellom brukerens nettleser og nettsidens backend.

2. **PHP Backend:** Backend av systemet er utviklet med PHP, et greit og fleksibelt skriptspråk for webutvikling. PHP er ansvarlig for å behandle server-side logikk og generere dynamisk innhold som sendes tilbake til klienten.

3. **MariaDB Database:** Som databaseløsning bruker systemet MariaDB, en av MySQL-relasjonsdatabasesystemet. MariaDB håndterer lagring og henting av data, og det er viktig for at applikasjonen skal kunne utføre effektive databaseoperasjoner.

4. **Azure Hosting:** Hele systemet er vert på Microsoft Azure-plattformen, som gir en skyinfrastruktur for å sikre skalerbarhet, høy tilgjengelighet og en rekke tjenester for å optimalisere driftsmiljøet.

Kommunikasjon og Samhandling:

- Webserveren mottar brukerforespørsler og dirigerer dem til den relevante PHP-backend-koden.
- PHP-koden samhandler med MariaDB-databasen for å hente, lagre og vise data.
- Systemet drar nytte av Azure-plattformens tjenester for å optimalisere ytelse og sikkerhet.

2. Backend

Beskrivelse av den overordnede arkitekturen

Min backend-arkitektur fungerer på følgende måte: Når en bruker ønsker å opprette en ny konto, blir informasjonen sendt som POST-kommandoer gjennom HTTP. Disse kommandoene blir deretter videresendt til databasen på serveren min, hvor de blir behandlet og lagret. Når brukeren senere besøker nettsiden, blir dataen hentet fra databasen og dynamisk presentert på nettsiden. Denne arbeidsflyten gjør det mulig for brukere å samhandle med nettsiden, legge til eller hente informasjon, og sikrer samtidig at dataen er persistent og tilgjengelig for brukere ved behov.

3. Serverkonfigurasjon

Operativsystem og versjon

Serveren bruker operativsystemet Linux Debian 12. Valget av Debian 12 som operativsystem for servermiljøet er fordi det er mer stabilt. Debian's fleksibilitet og tilpasningsmuligheter gir meg full kontroll over serverkonfigurasjonen, mens det aktive fellesskapet og engasjementet for fri og åpen kildekode sikrer solid støtte og en bred base av kunnskap. Debian 12 er optimalisert for å opprettholde en trygg, skalerbar og tilpasset serverinfrastruktur i samsvar med min behov og prinsipper.

Serverprogramvare

For å sikre en velfungerende nettside og database har jeg konfigurert Apache og MySQL-server på serveren. Apache fungerer som webserver og håndterer HTTP-forespørsler, mens MySQL tar seg av lagring og håndtering av data. Denne kombinasjonen gir en robust infrastruktur som støtter pålitelig hosting og effektiv administrasjon av nettsiden og tilhørende database.

Nettverks og sever konfigurasjon

For å etablere en Linux-server med en virtuell maskin (VM), begynner du med å opprette en konto på Azure eller logge deg inn med en eksisterende skole- eller jobbkonto. Deretter oppretter du en ny VM med prosjektnavnet som identifikator. Ved å tilpasse VM-konfigurasjonen i samsvar med prosjektets krav inkludert viktige aspekter som operativsystem, minne og prosessorkraft - sørg for å inkludere en offentlig IP-adresse for ekstern tilgang.

For å administrere serveren og vise nettsiden, bruker du den tildelte offentlige IP-adressen for å logge deg på VM-en via SSH og besøke nettsiden gjennom nettleseren. Dette gir deg tilgang til selve servermiljøet og visning av nettsiden eksternt.

Når serveren er satt opp, er det viktig å konfigurere Apache, webserverprogramvaren. Dette gjøres ved å installere og konfigurere Apache for å håndtere HTTP-forespørsler. Videre konfigurerer du MySQL-serveren for å håndtere lagring og administrasjon av databasen for prosjektet ditt.

For Apache-konfigurasjonen, tilpass HTTP-serverens innstillinger, inkludert virtual hosts, filrettigheter og sikkerhetsforanstaltninger for å sikre pålitelig levering av nettsideinnhold.

Når det gjelder MySQL-serveren, sørger du for å implementere sikre tilkoblinger, konfigurere brukerrettigheter og administrere databasen for optimal ytelse. Dette innebærer å opprette nødvendige databaser, tabeller og utføre annen konfigurasjon som kreves for prosjektets spesifikasjoner.

I nettverksdiagrammet bør du illustrere forbindelsen mellom Azure-kontoen, VM-en, offentlig IP-adresse, SSH-tilkobling og tilgang til nettsiden. Dette gir en visuell representasjon av den opprettede infrastrukturen, og gir en klar forståelse av hvordan alle komponentene samhandler.

4. Utviklingsmiljø

Utviklingsmiljøet

Utviklingsmiljøet er satt opp for å støtte effektiv og strukturert programvareutvikling. Jeg bruker en kombinasjon av lokale utviklingsmaskiner og skybaserte ressurser for å muliggjøre fleksibilitet og samarbeid. Utviklingsmiljøet er konfigurert med nødvendige verktøy og avhengigheter for å lette en jevn utviklingsprosess.

Verktøy og IDE-er som brukes

Jeg foretrekker Visual Studio Code som mitt hoved-IDE (Integrated Development Environment) for sin letthet, utvidbarhet og omfattende støtte for et bredt spekter av programmeringsspråk. Andre verktøy inkluderer Git for versjonskontroll.

Konvensjoner og beste praksis for koding

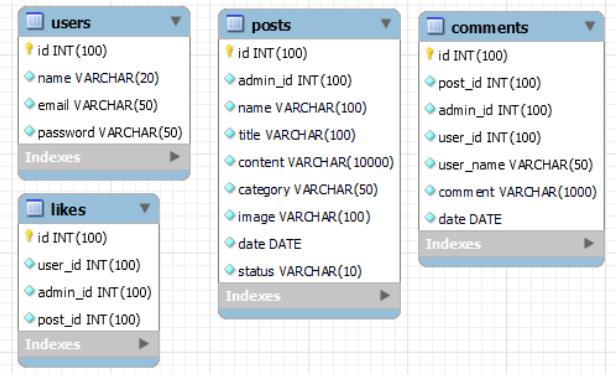
For å opprettholde en konsistent og lesbar kodebase, følger jeg et sett med konvensjoner og beste praksis for koding. Dette inkluderer men er ikke begrenset til bruk av deskriptive variabelnavn, konsistent innrykk og formatering, samt dokumentasjon av koden gjennom kommentarer. Dette gir ikke bare et ryddig og forståelig kodegrunnlag.

5. Databaseadministrasjon

Databaseadministrasjonsverktøy

For å administrere databasen min har jeg benyttet terminalen ved å koble meg til MariaDB via kommandolinjen ved å bruke "mysql -u Dawid -p". Gjennom denne tilnærmingen har jeg muligheten til å utføre operasjoner som å legge til, fjerne og endre data direkte fra kommandolinjegrensesnittet. Dette gir en effektiv og direkte måte å håndtere databaseoperasjoner på uten å måtte stole på grafiske grensesnitt.

Database



7. Deployering og Oppdatering

Nettsiden er distribuert gjennom Apache på serveren. Når jeg må gjøre endringer eller oppdateringer på nettsiden, følger jeg denne prosessen: Jeg utfører endringene lokalt, pusher dem deretter til GitHub. Deretter fjerner jeg hoveddirectory fra serveren og overfører den oppdaterte directory fra Windows-maskinen til serveren. Dette sikrer en smidig og strukturert oppdateringsprosess, og gir meg kontroll over kodeendringer fra lokale utviklingsmiljøer til den deployede nettsiden på serveren.