# Database relatie beschrijving

**Class** : Onderzoek

**Attributen** : Titel(string), beschrijving(string), Datum(DateTime), locatie(string), status(string)[codes voor gesloten, in behandeling etc?]

**Relaties**:

Onderzoek heeft 1 of meerdere uitvoerders (Uitvoerder=bedrijf, list van Bedrijf objecten?)

Onderzoek is van type SoortOnderzoek(Onderzoektype)

Onderzoek heeft meerdere Deelnemers(ervaringsdeskundige, lijst van Ervaringsdeskundige[type gebruiker?])

Onderzoek heeft betrekking op 1 of meerdere Beperkingen(List<Beperking>)

**Class** : Bedrijf

**Attributen** : bedrijfID(int), Naam(string), websitelink(string), locatie(string), beschrijving(string)

**Relaties**:

Bedrijf heeft 1 of meerdere contactpersonen(List<Contactpersoon>)

**Class** : Gebruiker (basisklasse voor ervaringsdeskundige en contactpersoon bedrijf?)

**Attributen** : GebruikerID(int), Voornaam(string), achternaam(string), emailadres(string), telefoonnummer(int)

*Beschikbare tijden?*

Relaties:

**Class** : Ervaringsdeskundige(Gebruiker)

**Attributen** : GebruikerID(PK, FK naar Gebruiker),Postcode(string), BenaderingTelefonisch(boolean), BenaderingPortal(boolean), CommercieleBenadering(boolean)

**Relaties**:

Ervaringsdeskundige heeft 1 of meerdere beperkingen(list van beperkingen)

Ervaringsdeskundige heeft 1 of meerdere hulpmiddelen (list van hulpmiddelen)

Ervaringsdeskundigen heeft geen of meerdere type\_onderzoek voorkeuren

**Class** : Beperking

**Attributen** : BeperkingsID(int), Naam(string)

**Relaties**:

**Class** : Hulpmiddelen

**Attributen** : Naam (String), Hulpdoel(string)

**Methodes** :

Relaties:

**Class** : OnderzoekSoort

**Attributen** : OnderzoekSoortId(int), Type\_onderzoek(string)

**Methodes** :

**Relaties**:

**Class** : Contactpersoon(Gebruiker)

**Attributen** : GebruikerID(PK, FK naar Gebruiker), Titel(string)

**Methodes** :

**Relaties**:

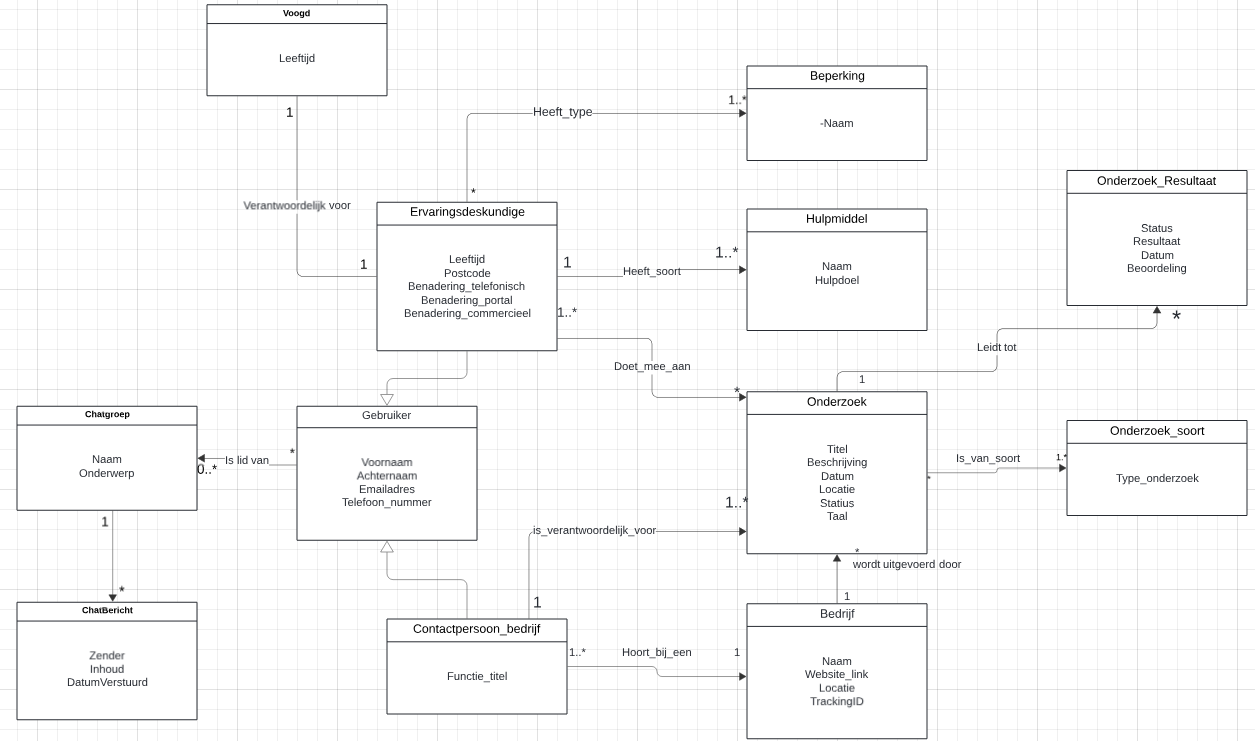
Class :

Attributen :

Methodes :

Relaties:

# Klassendiagram



# Relationeel Representatie Model

Het relationeel representatiemodel is opgebouwd met het oog op het organiseren van de data op een gestructureerde en begrijpelijke manier. In dit model worden de essentiële entiteiten (ofwel attributen) en hun onderlinge relaties vastgelegd. Bij het ontwerpen van dit model hebben we ook normalisatie toegepast om de consistentie, efficiëntie en integriteit van de gegevens te waarborgen.

Een belangrijk onderdeel van het representatiemodel, in vergelijking tot het bovenstaande klassendiagram, is het introduceren van primaire en vreemde sleutels. Dit zorgt voor unieke identificatie van entiteiten binnen een tabel en het definiëren van relaties tussen diverse tabellen. Het gebruik van samengestelde, zoals bij *Contactpersoon\_bedrijf,* biedt een manier om de relatie tussen entiteiten vast te leggen. Daarnaast zijn attributen zoals *Benadering\_telefonisch, Benadering\_portal* en *Commerciele\_benadering* als afzonderlijke velden gemodelleerd om de verschillende benaderingsmethoden van een Ervaringsdeskundige te registreren.

Een belangrijk onderdeel bij het modelleren van een representatiemodel is het normaliseren van een database structuur. Normalisatie is een cruciaal proces om de gegevensstructuren te optimaliseren, redundantie te minimaliseren en de integriteit van de database te waarborgen. Door normalisatie wordt de database efficiënter en wordt het risico van inconsistente gegevens verminderd.

Een belangrijke normalisatiestap in dit model is het gebruik van unieke identificatoren als primaire sleutels, zoals *Gebruiker\_ID, Bedrijf\_ID, en Onderzoek\_ID*. Dit helpt om ‘duplicates’ te voorkomen en biedt een gestandaardiseerde manier om entiteiten te identificeren. Daarnaast worden samengestelde sleutels, zoals die in Contactpersoon\_bedrijf, gebruikt om complexe relaties tussen entiteiten op een duidelijke manier weer te geven.

Het scheiden van entiteiten in verschillende tabellen, zoals *Gebruiker*, *Ervaringsdeskundige*, *Bedrijf*, en *Onderzoek*, is een verdere normalisatiestap. Hierdoor wordt voorkomen dat gegevens onnodig worden herhaald en wordt de database gestroomlijnd voor efficiënt beheer en onderhoud.

**Gebruiker(Gebruiker\_ID, Voornaam, Achternaam, Emailadres,Telefoonnummer),**

Gebruiker\_ID is Primary Key

**Ervaringsdeskundige(Ervaringsdeskundige\_ID, postcode, benadering \_ telefonisch, benadering\_portal, benadering\_commercieel)**

Ervaringsdeskundige\_ID is Primary Key

**Contactpersoon\_bedrijf(*Gebruiker\_ID*, functie\_titel, *bedrijf\_id*)**

Gebruiker\_ID is Primary Key

Gebruiker\_ID is Foreign Key, (Gebruiker\_Id in Gebruiker) NOT NULL

*Bedrijf\_id* is Foreign Key, verwijst naar Bedrijf(bedrijf\_id) NOT NULL

**Beperking(beperking\_ID, naam)**

Beperking\_ID is Primary Key

**Hulpmiddel(hulpmiddel\_ID, naam, hulpdoel)**

Hulpmiddel\_id is Primary Key

**Onderzoek(Onderzoek\_ID, Titel, Beschrijving, Datum, Locatie, Status, *bedrijf\_id, gebruiker\_id*)**

Onderzoek\_ID is Primary Key

*Bedrijf\_id* is Foreign Key, verwijst naar Bedrijf(bedrijf\_id) NOT NULL

*Gebruiker\_id* is Foreign Key, verwijst naar Contactpersoon\_bedrijf(gebruiker\_id) NOT NULL

**Onderzoek\_soort(onderzoek\_soort\_nr, type\_onderzoek)**

Onderzoek\_soort\_nr is Primary Key

**Onderzoek\_Van\_Soort(*Onderzoek\_ID, Onderzoek\_Soort\_Nummer*)**

Onderzoek\_ID, Onderzoek\_Soort\_Nummer is Primary Key

Onderzoek\_ID is Foreign Key, verwijst naar Onderzoek\_ID in (Onderzoek) NOT NULL

Onderzoek\_Soort\_Nummer is Foreign Key, Onderzoek\_Soort\_Nummer in (Onderzoek\_Soort) NOT NULL

**Onderzoek\_resultataat(Onderzoek\_resultaat\_id, status, resultaat, datum, beoordeling, *onderzoek*)**

Onderzoek\_resultaat\_id is Primary Key

Onderzoek is Foreign Key, verwijst naar Onderzoek(onderzoek\_id), NOT NULL

**Deelname(*Onderzoek\_ID, Ervaringsdeskunde\_ID*, datum)**

**Bedrijf(bedrijf\_ID, bedrijfs\_naam, website\_link, locatie)**

Bedrijf\_ID is Primary Key

**Ervaringsdeskundige\_Beperking(ervaringsdeskundige\_ID,,beperking\_ID)**

ervaringsdeskundige\_ID,,beperking\_ID is Primary Key

ervaringsdeskundige\_ID, is Foreign Key verwijst naar Ervaringsdeskundige (ervaringsdeskundige\_ID) NOT NULL

beperking\_ID is foreign key verwijst naar Beperking (beperking\_ID) NOT NULL

**ChatGroep(ChatGroepId, Naam, onderwerp)**  
ChatGroepId is Primary Key

**ChatBericht(ChatBerichtId, *Verzender*, Inhoud, DatumVerstuurd)**

ChatBerichtId is Primary Key

Verzender is Foreign Key, verwijst naar Gebruiker(GebruikerId)

**ChatGroep\_ChatBericht(*ChatGroepId, ChatBerichtId*)**

ChatGroepId, ChatBerichtId is Primary Key

ChatBerichtId is Foreign Key, verwijst naar ChatBericht(ChatBerichtId)

ChatGroepId is Foreign Key, verwijst naar ChatGroep(ChatGroepId)

# Relationeel Implementatiemodel

Het relationeel implementatiemodel vertraal het representatiemodel naar SQL-tabellen, waarbij de structuur wordt omgezet in een (uiteindelijke) fysieke database. Bij het implementeren van het model zijn enkele keuzes gemaakt om de entiteiten (ofwel attributen) en relaties om te zetten in databaseconstructies (zie ommezijde, *pagina 7*).

Auto-incrementing identiteitsvelden (zoals *onderzoeks\_id, gebruiker\_id etc)* zijn ingevoerd om unieke waarden toe te wijzen en de gegevensintegriteit te waarborgen. Het gebruik van externe sleutels, zoals in *Onderzoeks-tabel*, zorgt voor referentiële integriteit en legt de relaties tussen de verschillende tabellen vast.

De keuze voor het gebruik van bits, zoals Benadering\_telefonisch, in het Ervaringsdeskundige-schema biedt een gestandaardiseerde manier om Boolean-waarden op te slaan, wat nuttig is voor efficiëntie en eenvoudige interpretatie van gegevens.

De implementatie volgt de principes van databasenormalisatie door redundante gegevens te vermijden en consistente relaties tussen tabellen te behouden.

**Onderzoek:**

| CREATE TABLE Onderzoek ( Onderzoek\_id INT IDENTITY(1,1) NOT NULL PRIMARY KEY, titel VARCHAR (255), beschrijving VARCHAR(255), datum DATE, locatie VARCHAR(255), status VARCHAR(255), bedrijf INT NOT NULL, contactpersoon INT NOT NULL,  FOREIGN KEY(bedrijf) REFERENCES Bedrijf(bedrijf\_id), FOREIGN KEY(contactpersoon) REFERENCES Contactpersoon\_bedrijf(gebruiker\_id) ); |
| --- |

**Onderzoeks\_Soort :**

| CREATE TABLE Onderzoeks\_Soort(  onderzoek\_soort\_id INT IDENTITY(1,1) NOT NULL,  type\_onderzoek VARCHAR(255),  PRIMARY KEY(onderzoek\_soort\_id)  ); |
| --- |

**Onderzoek\_Van\_Soort :**

| CREATE TABLE Onderzoek\_Van\_Soort(  Onderzoek\_id INT,  Onderzoek\_soort\_id INT,  PRIMARY KEY (onderzoek\_id, onderzoek\_soort\_id),  FOREIGN KEY (onderzoek\_id) REFERENCES Onderzoek(onderzoek\_id) ON DELETE CASCADE  FOREIGN KEY (onderzoek\_soort\_id) REFERENCES Onderzoeks\_soort(onderzoek\_soort\_id) ON DELETE CASCADE ) |
| --- |

**Bedrijf**:

| CREATE TABLE Bedrijf (  bedrijf\_id INT IDENTITY(1,1) NOT NULL,   naam VARCHAR(255),   websitelink VARCHAR(255),   locatie VARCHAR(255),  PRIMARY KEY(bedrijf\_id) ); |
| --- |

**Gebruiker:**

| CREATE TABLE Gebruiker(  gebruiker\_id INT IDENTITY(1,1) NOT NULL,  voornaam VARCHAR(255),  achternaam VARCHAR(255),  emailadres VARCHAR(255) UNIQUE,  telefoonnummer VARCHAR(255) UNIQUE,    PRIMARY KEY(gebruiker\_id) ); |
| --- |

**Ervaringsdeskundige :**

| CREATE TABLE Ervaringsdeskundige ( ervaringsdeskundige\_ID INT IDENTITY(1,1) NOT NULL, postcode VARCHAR(255), benadering\_telefonisch bit, benadering\_portal bit, commerciele benadering bit, PRIMARY KEY(ervaringsdeskundige\_id) ); |
| --- |

**Beperking:**

| CREATE TABLE Beperking( beperking\_id INT IDENTITY(1,1) NOT NULL, naam VARCHAR(255),  PRIMARY KEY (beperking\_id) ); |
| --- |

**Hulpmiddel:**

| CREATE TABLE Hulpmiddel(  hulpmiddel\_id INT IDENTITY(1,1) NOT NULL,  naam VARCHAR(255),   hulpdoel VARCHAR(255),  PRIMARY KEY(hulpmiddel\_id) ); |
| --- |

**Ervaringsdeskundige\_Beperking**

| CREATE TABLE Ervaringsdeskundige\_Beperking (  deskundige\_id INT,  beperking\_id INT,  PRIMARY KEY(gebruiker\_id, beperking\_id),  FOREIGN KEY (gebruiker\_id) REFERENCES Ervaringsdeskundige(gebruiker\_id) ON DELETE CASCASE  FOREIGN KEY (beperking\_id) REFERENCES Beperking(beperking\_id) ON DELETE CASCADE ) |
| --- |

**Ervaringsdeskundige\_Hulpmiddel**

| CREATE TABLE Ervaringsdeskundige\_Hulpmiddel (  Deskundige\_id INT,  Hulpmiddel\_id INT ,  PRIMARY KEY(gebruiker\_id, beperking\_id),  FOREIGN KEY (gebruiker\_id) REFERENCES Ervaringsdeskundige(gebruiker\_id) ON DELETE CASCASE  FOREIGN KEY (hulpmiddel\_id) REFERENCES Hulpmiddel(hulpmiddel\_id) ON DELETE CASCADE ) |
| --- |

**Contact\_persoon**:

| CREATE TABLE Contact\_persoon( gebruiker\_id INT IDENTITY(1,1) NOT NULL, bedrijf\_id INT,  titel VARCHAR(255),  PRIMARY KEY (gebruiker\_id),  FOREIGN KEY (gebruiker\_id) REFERENCES Gebruiker(gebruiker\_id) NOT NULL, FOREIGN KEY (bedrijf\_id) REFERENCES Bedrijf(bedrijf\_id) NOT NULL  ); |
| --- |

**Onderzoek\_resultaat:**

CREATE TABLE Onderzoek\_resultaat (

onderzoek\_resultaat\_id *INT* IDENTITY(1,1) NOT NULL,

status *VARCHAR*(255),

resultaat(*VARCHAR*(255),

datum *DATE*,

beoordeling *VARCHAR*(255)

onderzoek *INT*,

*PRIMARY KEY* (onderzoek\_resultaat),

*FOREIGN KEY*(onderzoek) *REFERENCES* Onderzoek(onderzoek\_id) NOT NULL

);

**ChatGroep**

| CREATE TABLE ChatGroep(  ChatGroepId INT Identity(1,1) NOT NULL,  Naam varchar(100) NOT NULL,  Onderwerp varchar(255) NOT NULL,  PRIMARY KEY (ChatGroepId) ) |
| --- |

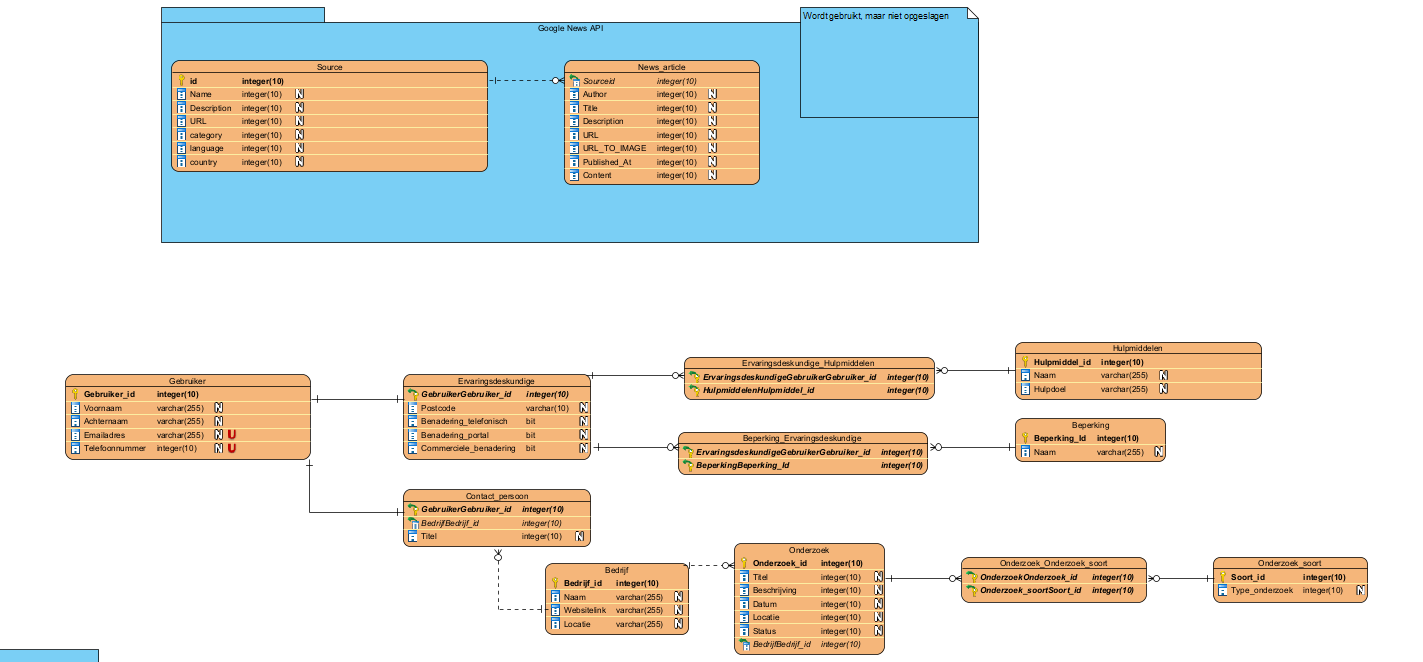
**ChatBericht**

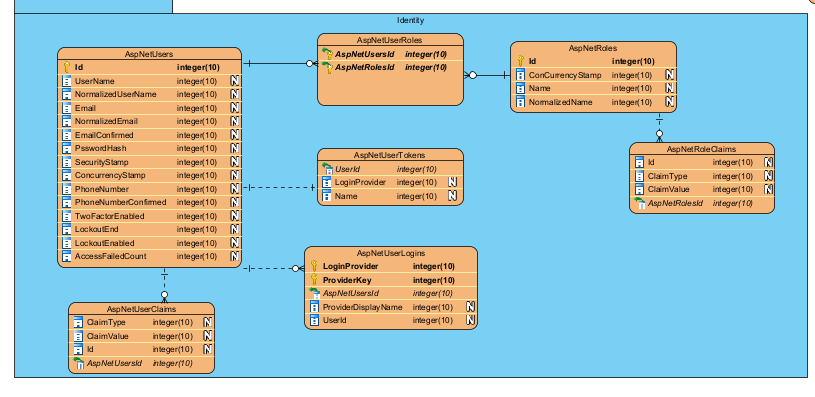
| CREATE TABLE ChatBericht(  ChatBerichtId INT Identity(1,1) NOT NULL,  Content varchar(1000) NOT NULL,  Date DatumVerstuurd NOT NULL,  Verzender INT NOT NULL,  PRIMARY KEY(ChatBerichtId),  FOREIGN KEY(Verzender) REFERENCES Gebruiker(GebruikerId) NOT NULL ) |
| --- |

**ChatGroep\_ChatBericht**

| CREATE TABLE ChatGroep\_ChatBericht(  ChatGroepID INT NOT NULL,  ChatBerichtID INT NOT NULL,  PRIMARY KEY(ChatGroepId, ChatBerichtId),  FOREIGN KEY(ChatGroepId) REFERENCES ChatGroep(ChatGroepId) NOT NULL,  FOREIGN KEY(ChatBerichtID) REFERENCES ChatBericht(ChatBerichtId) NOT NULL ) |
| --- |

**Datamodel**

****

****

# API’s

* Google News API(voor berichten over accessibility)
* Google Sign In API
* *SignalR API?*
* LinkedIn’s AutoFill Plugin (Om profile form fields automatisch in te vullen met hun gegevens van LinkedIn)
* ASP.NET Core Identity? (voor gebruikersauthenticatie en autorisatie)
* Swagger/OpenAPI? (voor het documenteren van onze API, zodat deze beter te begrijpen is voor de stichting Accessibility)
* Axe Core API (om de toegankelijkheid van de website te testen)

# Identity Providers

Microsoft

Google

LinkedIn plugin

# Clickstream

De bedoeling is om een script te schrijven dat op een website van een bedrijf geplaatst kan worden. Dit script verzamelt informatie over hoe de interactie tussen klant en website gaat en stuurt dit naar de backend van de developers.

* Het is eenvoudiger om zelf een script te maken dan om google analytics te gebruiken.

Aparte diagrammen voor dataopslag en data-interactie

Identitytabellen en API informatie in database tabellen

ERD acceptabel, UML heeft voorkeur

Code diagrammen implementeren (sequence diagrams voor bijvoorbeeld API-relaties)

Clickstream script zelf schrijven, niet rondklooien met Google Analytics

# 

# 

# 

# 

# 

# Literatuurlijst

Das, T. (2023, October 26). *How ClickStream Data can tell what Google Analytics cannot*. Geekflare. <https://geekflare.com/clickstream-data/> )

Mukhtar, S. (2004, August 23). *Applying Robustness Analysis on the Model–View–Controller (MVC) Architecture in ASP.NET Framework, using UML*. CodeProject. <https://www.codeproject.com/Articles/8058/Applying-Robustness-Analysis-on-the-Model-View-Con>

*Review of UML class Diagram of a ASP.NET Core Web API*. (n.d.). Stack Overflow. <https://stackoverflow.com/questions/70736632/review-of-uml-class-diagram-of-a-asp-net-core-web-api>

*Database modeling with UML | SPARX Systems*. (z.d.). <https://sparxsystems.com/resources/tutorials/uml/datamodel.html>

Uml voor de chatfunctionaliteit:

<https://creately.com/diagram/example/i8d5u68p3/uml-class-diagram-for-chat-application>

Booleans(bits) in SQL

<https://www.sqlshack.com/sql-boolean-tutorial/>

freeCodeCamp.org. (2018, 10 februari). How to track user interactions in your React App.

<https://www.freecodecamp.org/news/how-to-track-user-interactions-in-your-react-app-b82f0bc4c7ff/>

Brightspace Course over databases:

<https://brightspace.hhs.nl/d2l/home/56629>

Derks, Brigitte(Versie 2023-2024). *Reader DB-ontwerp H-SE-S3DB*

<https://brightspace.hhs.nl//content/enforced/56629-H-SE-S3DB-1-21_2023_VT/Reader%20DB-ontwerp%202023-20243.pdf?_&d2lSessionVal=aebR1ILu6dopFAOSNX0jHVYGB&ou=56629>

Jovanpop-Msft. (2023, March 10). *Store JSON documents in SQL Server or SQL Database - SQL Server*. Microsoft Learn. <https://learn.microsoft.com/en-us/sql/relational-databases/json/store-json-documents-in-sql-tables?view=sql-server-ver16>

<https://code-maze.com/efcore-store-json-in-an-entity-field/>