.NET Webform: Member Skills & Qualifications Template

Created: 2025/05/07, 21:22:14
Last Updated: 2025/05/07, 21:22:39

User

I have a aspx .net 4.8.1 webform with c# and Bootstrap 5

I have a members_all table and the members has an occupation, skill and qualifications. I need a template per memberID to control these

I suggest a MememberCapabilities tables with there memberid, ,the type of skill and a ranking of 1-10 per skill, also paid for skill or non paid. And a column for their offer to the church for probona.

We need to have categories for 0ccupation/Skills/Qualifications/Level of qualification (like Matric/diploma/degree)

Then unemployed as a cat.

each must be ranked.

the members are in members_all.memberid and we will then have a skills table and memberskill join, with date ranges and if active, archived

---

Assistant (gemini-2.5-flash-thinking)

Right, Lee-Anne, that sounds like a sensible way to structure the information! Breaking down members' capabilities into categories like Occupation, Skills, and Qualifications, and then tracking them individually with rankings and status (like paid/unpaid skills or active/archived entries) will give you a great deal of flexibility.

Based on your suggestions and the need for categorisation, here's a proposed database schema structure that builds on your ideas:

We'll use a few tables to keep the data organised and maintain relationships correctly.

1. `members_all`
    - This table already exists and contains your core member information, including `memberid`. This will be the central table that the others link back to.

```
-- Example structure, assuming this exists
CREATE TABLE members_all (
    memberid INT PRIMARY KEY,
    -- ... other member fields ...
);
```

2. `CapabilityCategory`

   - This table will define the broad types of capabilities.
   - Examples: 'Occupation', 'Skill', 'Qualification', 'Qualification Level', 'Pro Bono Offer'.

```
CREATE TABLE CapabilityCategory (
    CategoryID INT IDENTITY(1,1) PRIMARY KEY,
    CategoryName NVARCHAR(100) NOT NULL UNIQUE
);
```

   - *Initial Data Examples:*

```
INSERT INTO CapabilityCategory (CategoryName) VALUES
('Occupation'),
('Skill'),
('Qualification'),
('Qualification Level'),
('Pro Bono Offer');
```

3. `Capability`

   - This table will hold the specific capabilities within each category.
   - Examples: 'Carpenter' (under Occupation), 'SQL' (under Skill), 'Matric' (under Qualification), 'Degree' (under Qualification Level), 'Website Design' (under Skill, potentially linked to 'Pro Bono Offer' via `MemberCapability`). You could also have a 'CapabilityName' like 'Unemployed' linked to the 'Occupation' category here.

```
CREATE TABLE Capability (
    CapabilityID INT IDENTITY(1,1) PRIMARY KEY,
    CategoryID INT NOT NULL,
    CapabilityName NVARCHAR(255) NOT NULL,
    -- Optional: Description NVARCHAR(MAX) NULL,
    CONSTRAINT FK_Capability_Category FOREIGN KEY (CategoryID)
        REFERENCES CapabilityCategory (CategoryID)
);
```

   - *Initial Data Examples (after populating `CapabilityCategory`):*

```
-- Assuming CategoryID 1 is 'Occupation', 2 is 'Skill', 3 is
'Qualification', 4 is 'Qualification Level'
INSERT INTO Capability (CategoryID, CapabilityName) VALUES
(1, 'Unemployed'),
(1, 'Carpenter'),
(1, 'Teacher'),
```

```
(2, 'SQL'),
(2, 'Project Management'),
(3, 'Matric'),
(3, 'Diploma'),
(3, 'Degree'),
(4, 'NQF Level 4'), -- Matric
(4, 'NQF Level 6'), -- Diploma
(4, 'NQF Level 7'); -- Degree
-- You'd add more specific skills, qualifications, etc.
```

4. `MemberCapability`
    - This is the join table that links members (`members_all`) to specific capabilities (`Capability`) and includes details specific to that member's relationship with that capability.
    - This table incorporates the ranking, paid status, pro bono offer details, date ranges, and active status you mentioned.

```
CREATE TABLE MemberCapability (
    MemberCapabilityID INT IDENTITY(1,1) PRIMARY KEY,
    MemberID INT NOT NULL,
    CapabilityID INT NOT NULL,
    Ranking INT NULL CHECK (Ranking >= 1 AND Ranking <= 10), --
Optional, specifically for skills (1-10)
    IsPaidSkill BIT NULL, -- Optional, TRUE/FALSE for skills
    ProBonoOfferDetails NVARCHAR(MAX) NULL, -- Text details for a
Pro Bono offer entry
    StartDate DATE NULL, -- When the member gained/was active in
this capability
    EndDate DATE NULL, -- When the member stopped being active in
this capability
    IsActive BIT NOT NULL DEFAULT 1, -- 1 for Active, 0 for
Archived/Inactive
    DateAdded DATETIME NOT NULL DEFAULT GETDATE(),
    DateModified DATETIME NULL,
    CONSTRAINT FK_MemberCapability_Member FOREIGN KEY (MemberID)
        REFERENCES members_all (memberid),
    CONSTRAINT FK_MemberCapability_Capability FOREIGN KEY
(CapabilityID)
        REFERENCES Capability (CapabilityID)
);
```

How this structure addresses your points:

- **Template per MemberID:** By querying the `MemberCapability` table filtering by `MemberID`, you can retrieve all capabilities linked to a specific member. Your Web Form page for a member would fetch and display these entries, effectively creating their capability "template".
- **Categories:** The `CapabilityCategory` and `Capability` tables provide a structured way to define and categorise all possible occupations, skills, qualifications, and levels.

- **Ranking:** The `Ranking` column in `MemberCapability` allows for scoring skills (1-10) as you requested. It's nullable, as not all capabilities (like qualifications) will have a ranking.
- **Paid/Non-Paid:** The `IsPaidSkill` column handles this for skills. It's also nullable, only applicable where relevant.
- **Pro Bono Offer:** You can have a `CapabilityCategory` called 'Pro Bono Offer'. Specific types of pro bono offers (e.g., 'Website Design', 'Gardening Help') could be entries in the `Capability` table linked to this category. The `ProBonoOfferDetails` field in `MemberCapability` allows the member to add specific text details for that particular offer.
- **Unemployed:** Create a `Capability` entry called 'Unemployed' under the 'Occupation' category. A member can then link to this capability in `MemberCapability` and mark it as `IsActive=TRUE`.
- **Date Ranges and Active/Archived:** The `StartDate`, `EndDate`, and `IsActive` columns in `MemberCapability` allow you to track the timeline and current status of each specific capability entry for a member.

Web Forms Implementation (`.aspx` and C#):

On your ASP.NET Web Form page for a member:

1. Fetch the `MemberID` (e.g., from the URL or a session variable).
2. Query the `MemberCapability` table, joining with `Capability` and `CapabilityCategory`, filtering by the `MemberID`.
3. Group the results by `CategoryName` (`Occupation`, `Skill`, `Qualification`, etc.).
4. Display the list of capabilities under each category for the member. You could use Bootstrap Cards, Accordions, or just structured `<div>` elements.
5. For adding/editing capabilities, you would have a form:
    - Dropdowns bound to the `CapabilityCategory` and `Capability` tables. The `Capability` dropdown would dynamically update based on the selected category using JavaScript or an UpdatePanel.
    - Input fields for `Ranking`, `IsPaidSkill`, `ProBonoOfferDetails`, `StartDate`, `EndDate`. These fields could be conditionally shown/hidden based on the selected `CapabilityCategory` or `Capability` type.
    - A checkbox for `IsActive`.
    - Buttons to Save (Insert/Update `MemberCapability`) or Delete.

This structure provides a clear, normalised way to manage member capabilities and should integrate well with your existing .NET 4.8.1 Web Forms application and Bootstrap 5 styling. Good luck with the implementation! Let me know if you have more questions.
