**Computer Science Project**

**Deliverable 1 (Started -> 14/08/2025):**

**Investigating (14/08/2025):**

**Article 1: *Volunteer Management For Nonprofits: Strategies, Challenges, and Best Practices***

* Article basically outlines why effective volunteer management is crucial, and shares the practical strategies for building sustainable programs

**Importance of Community Engagement:**

* Volunteers are needed, as they expand the nonprofits impact, boosts community service, and enhances credibility, all while keeping the costs of operation lower
* Effective program motivates volunteers, improves outcomes, and may help reduce staff turnover
* If volunteers’ skills & interests aligned with needs of organisation, it increases their productivity and fosters more volunteer engagement, also benefiting the volunteer themselves by allowing them to gain real-world experience

**Common Challenges:**

* Nonprofit organisations often face staff, budget, and time constraints associated with effective volunteer management
  + *Solution:* Investing in a volunteer management system can help to optimise these available resources
* Nonprofits can often get unorganised, and volunteers are not too sure what they are signing up for
  + *Solution:* Explain the organisation’s mission, and explain how volunteers can contribute to achieving that mission
* Establish roles, responsibilities, expectations for volunteers, and set measurable goals
* Non-profit organisations often have trouble attracting volunteers, and often lack positions that a diverse range of people can volunteer in
  + *Solution:* Create engaging volunteer opportunities that match the needs of potential volunteers
* Use a variety of channels (E.g. Social media, Local partnerships & events) to attract volunteers

**Reference:**

* NGOFeed. (2025). Volunteer Management for Nonprofits: Strategies, Challenges, and Best Practices. [online] Available at: <https://ngofeed.com/blog/volunteer-management-for-nonprofit/>[Accessed 14 Aug. 2025].

**Article 2: *Overcoming Challenges in Volunteer Management***

* Article basically provides practical advice on how to tackle the common volunteer management challenges effectively

**Importance of Community Engagement:**

* Strengthens social bonds, where the community engagement fosters trust, cooperation, and relationships between volunteers, organisations, and the people
* Boosts volunteer retention, where volunteers feel like they’re a part of the community, and see the real-world impacts of their effects, encouraging them to stay long-term
* Communities that are engaged with organisations more likely to donate, advocate, and recruit others to join

**Common Challenges:**

* Scheduling conflicts, where everyone has different availabilities, hence making it “a nightmare trying to coordinate”
  + *Solution:* Using a scheduling tool, from simple spreadsheets to more advanced software, find something that works for your team
* Encourage colleagues to be transparent about time commitments, so the volunteers know what’s expected
* Dealing with burnout, where volunteers can get easily overwhelmed, especially if they are taking on too much or if they are simply new to the whole system
  + *Solution:* Set clear role boundaries, and encourage breaks and self-care
* Respect volunteers’ limits and avoid overloading them
* Communication gaps, where volunteers are often don’t communicate as much as they need to, which leads to an unorganised structure, and a broken flow
  + *Solution:* Be open to using multiple channels (E.g. Email, Text, Social Media)
* Send regular updates to volunteers to stay informed and connected

**Key Takeaways:**

* Volunteers boost nonprofits impact, community service, and credibility, while also lowering operational costs
* It also brings communities closer together, and strengthens social bonds
* Some issues related to volunteer management include volunteer burnout, communication gaps, scheduling conflicts, organisation, attracting volunteers, and money

**Deconstruction (14/08/2025):**

* Will have the four main tables: Organisations, Volunteers, Events, Skills
  + But these all have many-to-many relationships (Except organisation -> event, which is one-to-one), and hence they need junction tables to resolve the many-to-many relationships
* Will be listing these in relational notation (I didn’t know what this meant in the test 🙁)
* Need a user table to login to the account (Mentioned in-class today);

**User**(user\_id, email, password\_hash, phone\_number, role, created\_at)

**Volunteer**(volunteer\_id, *user\_id*, first\_name, last\_name, availability)

**Organisation**(organisation\_id, *user\_id*, name, description, address, website\_url)

**Event**(event\_id, *organisation\_id*, title, description, event\_date, location, max\_volunteers)

**Skills**(skill\_id, name, description)

**Relationships:**

* Organisation and Event
  + Is a one-to-many relationship (One organisation can host many events, but every event belongs to exactly one organisation, no collaborated organisations or anything)
  + No junction table or anything needed
* Event and Volunteer
  + Is a many-to-many relationship (Volunteers can register for multiple events, and events can have multiple volunteers)
  + Requires junction table: Volunteer\_Event
  + Can also have the signup date

**Volunteer\_Event**(*volunteer\_id*, *event\_id*, signup\_date) (Composite primary key)

* Volunteer and Skill
  + Is a many-to-many relationship (Volunteers can have many skills, each skill can belong to multiple volunteers)
  + Requires junction table: Volunteer\_Skill
  + Should also have the proficiency of the volunteer in the skill

**Volunteer\_Skill**(*volunteer\_id*, *skill\_id*, proficiency\_level) (Composite primary key)

* Event and Skill (Maybe, still thinking about it)
  + Is a many-to-many relationship (An event can require multiple skills, a skill can be required by multiple events)
  + Requires junction table: Event\_Skill

**Event\_Skill**(*event\_id*, *skill\_id*) (Composite primary key)

**Final List of Tables:**

**User**(user\_id, email, password\_hash, phone\_number, role, created\_at)

**Volunteer**(volunteer\_id, *user\_id*, volunteer\_first\_name, volunteer\_last\_name, volunteer\_availability)

**Organisation**(organisation\_id, *user\_id*, organisation\_name, organisation\_description, organisation\_address, organisation\_website\_url)

**Event**(event\_id, *organisation\_id*, event\_title, event\_description, event\_date, event\_location, max\_volunteers)

**Skills**(skill\_id, skill\_name, skill\_description)

**Volunteer\_Event**(*volunteer\_id*, *event\_id*, event\_signup\_date) (Composite primary key)

**Volunteer\_Skill**(*volunteer\_id*, *skill\_id*, volunteer\_proficiency\_level) (Composite primary key)

**Event\_Skill**(*event\_id*, *skill\_id*) (Composite primary key)

**Entity Relationship Diagram (18/08/2025, updated 21/08/2025):**

**Normalisation (19/08/2025, updated 21/08/2025):**

**Unnormalised Form (0NF):**

* Is the single table, with everything in it, but only in relational notation, calling it ‘Community\_Connect’

**Issues:**

* Non-Atomic values, where the skills\_required lists all the skills the volunteer needs (Hence is 1NF)
* Organisation details are repeated for every event, and volunteer information repeated for every signup, so data is redundant
* Update Anomaly Example: Updating an organisation’s phone number requires updating it in all of the records
* Insert Anomaly Example: Can’t add a new volunteer until they sign up for an event
* Delete Anomaly Example: Deleting an event of an organisation, if it is their only event, wipes out the entire organisation
* Table also has partial and transitive dependencies, but talked about more in 2NF and 3NF

**Community\_Connect**(user\_id, email, password\_hash, role, created\_at,

volunteer\_id, volunteer\_first\_name, volunteer\_last\_name, volunteer\_email, volunteer\_phone\_number, volunteer\_availability, volunteer\_required\_skills, signup\_date

organisation\_id, organisation\_name, organisation\_description, organisation\_contact\_email, organisation\_phone\_number, organisation\_address, organisation\_website\_url,

event\_id, event\_title, event\_description, event\_date, event\_location, max\_volunteers)

**First-Normal Form (1NF):**

* All values are now atomic, with no lists or nested data, where repeating groups are eliminated
* No actual change in the relational notation of the tables, but would see change with actual data, where for example, every skill required would be on separate row

**Issues:**

* Primary key is no longer unique, so is no longer a “primary” key
* The non-key attributes do not depend on the whole primary key, so need to split them up into their separate tables, and hence create new tables for the fields that are partially dependent
* Data is redundant, where organisation information, volunteer information, event information repeated for each required skill of the volunteer

**Second-Normal Form (2NF):**

* Tables are now split up, and all partial dependencies are removed

**Issues:**

* A lot of transitive dependencies, mainly with the email, where the volunteer\_email, and the organisation\_email, should depend on the user\_id, as they are the email of the user
* Need to also resolve the many-to-many relationships using junction tables

**User**(user\_id, password\_hash, role, created\_at)

**Volunteer**(volunteer\_id, *user\_id*, volunteer\_first\_name, volunteer\_last\_name, volunteer\_email, volunteer\_phone\_number, volunteer\_availability)

**Organisation**(organisation\_id, *user\_id*, organisation\_name, organisation\_description, organisation\_email, organisation\_phone\_number, organisation\_address, organisation\_website\_url)

**Event**(event\_id, *organisation\_id*, event\_title, event\_description, event\_date, event\_location, max\_volunteers)

**Skills**(skill\_id, skill\_name, skill\_description)

**Third-Normal Form (3NF):**

* No more transitive dependencies in the data
  + Moved email to the user table, and avoided repeating it in Organisation and Volunteer tables

**Issues:**

* Need to also resolve the many-to-many relationships using junction tables, but is a part of database design and not normalising the database
  + Will be later resolved in the database design phase

**User**(user\_id, email, password\_hash, phone\_number, role, created\_at)

**Volunteer**(volunteer\_id, *user\_id*, volunteer\_first\_name, volunteer\_last\_name, volunteer\_availability)

**Organisation**(organisation\_id, *user\_id*, organisation\_name, organisation\_description, organisation\_address, organisation\_website\_url)

**Event**(event\_id, *organisation\_id*, event\_title, event\_description, event\_date, event\_location, max\_volunteers)

**Skills**(skill\_id, skill\_name, skill\_description)

**Data Dictionary ():**