# Readme: **Access Management Application**

Pre requisites:

- A computer with .NET framework 4.5 installed
- Visual Studio or any other tool to build the application
- SQL Server Express or above
- Internet connection (to download package dependencies)

Initializing the database:

- The application uses a SQL Server database
- Setup a new (preferably) local or remote database. Make sure all required permissions are granted
- The source includes a 'Scripts' folder which has all the SQL scripts to setup the database. Run all the scripts

Setting up the application & running:

- Easiest way is to open the source code in Visual Studio and build
- The source code does not include binaries and NuGet packages, the system will download packages and build binaries on build
- The database connection and folder paths need to be set correctly in the App.config file
- Once successfully built, select the 'AccessManagement.Console' application as the Start-up project and run. Alternatively it can be run from the created exe file

## A Brief about the application:

1. The application has all the frameworks required to support the requirements, built as .NET dlls
2. No elaborate UI was built, a simple console application is there to run the application
3. Once run, the application gives options to [1] Initialize the database [2] Run daily reports [3] Log in to the system
    a. Initializing database is the first step. It reads the LDAP service & Access Point APIs (simulated) and populate the database with data
    b. Daily reports create department wise attendance and activity logs (and stores them locally, as SMTP had conflicts with firewall)
    c. Login to the system with credentials as stored in LDAP
4. There is an option to check access denied notification, but that doesn't do much
5. On login, it shows success/failure message
6. For departmental managers it gives additional options to [1] View logs [2] See access point details [3] Manage user permissions
    a. View log simply shows the log location
    b. Second options show a brief list of all access points as received from Access Point Hardware APIs
    c. Manager can grant different types of access to others
7. Since no real LDAP service of proprietary hardware APIs were available, the application simulates those behaviours with plain csv data stored in \Source\Data

Sample user data:
User: ac@co.com/Password#7
Manager: haack@co.com/Password#4

## Requirement gaps, issues & assumptions:

1. Since LDAP maintenance is not part of application, it is assumed that the LDAP has all the necessary details, and is functional all the time.
2. Same with Access Point hardware APIs, assumed they are well maintained and accessible all the time.
3. The organization structure is not defined clearly. Departments have managers? Do employees have managers, or department managers become their managers too? Groups can have managers? Any restriction on hierarchy level?
4. The activity log that would be sent to department managers, what exactly defines activity? It not clear. As of now, any kind of access (entry/exit/access deny) are assumed to be activity.
5. It is mentioned that off work access will be considered access violation. Now, what defines duty time/work hours? 9-5 Monday to Friday? Holidays? Leaves? Not defined clearly.
6. It is assumed that only one manager is possible per entity (e.g. department). For multiple managers, some change will be required.
7. Can employee be part of multiple groups? Assumed yes.
8. It has been assumed, that managers are admins by default. No special privileges been granted to any group/department (e.g. HR)
9. Employee Login has been assumed to be unique numeric key. Though more details are required for Employees (like LDAP domain, DOB etc) they have been ignored from application perspective.
10. Nothing has been mentioned for UI. For simplicity, a simple console application has been built, but the same core can be used for any other type of applications, like web, mobile etc.