AES-256 Authentication : (Using Database)

BackUps for Authentication (Master key Auth) :

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Espl@9999999999#';

CREATE CERTIFICATE UserAuthCertificate_Dev
   WITH SUBJECT = 'DataEncryptionCert_Dev2024';

CREATE SYMMETRIC KEY UserAuthSymmetricKey_Dev
   WITH ALGORITHM = AES_256
   ENCRYPTION BY CERTIFICATE UserAuthCertificate_Dev;
```

Now the Encryption and Encryption Procedures are as follows:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER   PROCEDURE [dbo].[EncryptPassword]
   @Password NVARCHAR(MAX),
   @EncryptedPassword VARBINARY(MAX) OUTPUT
AS
BEGIN
   DECLARE @SymmetricKey NVARCHAR(MAX) = 'UserAuthSymmetricKey_Dev';
   DECLARE @Certificate NVARCHAR(MAX) = 'UserAuthCertificate_Dev';

   BEGIN TRY
       OPEN SYMMETRIC KEY [UserAuthSymmetricKey_Dev]
           DECRYPTION BY CERTIFICATE [UserAuthCertificate_Dev];

       SET @EncryptedPassword = ENCRYPTBYKEY(KEY_GUID(@SymmetricKey), @Password);

       CLOSE SYMMETRIC KEY [UserAuthSymmetricKey_Dev];
   END TRY
   BEGIN CATCH

       SET @EncryptedPassword = NULL;
       PRINT 'Error: ' + ERROR_MESSAGE();
   END CATCH
END;
GO
```

```
ALTER PROCEDURE [dbo].[DecryptPassword]
   @EncryptedPassword VARBINARY(MAX),
   @PlainPassword NVARCHAR(MAX) OUTPUT
AS
BEGIN
   DECLARE @SymmetricKey NVARCHAR(MAX) = 'UserAuthSymmetricKey_Dev';
   DECLARE @Certificate NVARCHAR(MAX) = 'UserAuthCertificate_Dev';

   OPEN SYMMETRIC KEY [UserAuthSymmetricKey_Dev]
   DECRYPTION BY CERTIFICATE [UserAuthCertificate_Dev];

   SET @PlainPassword = CONVERT(NVARCHAR(MAX), DECRYPTBYKEY(@EncryptedPassword));

   CLOSE SYMMETRIC KEY [UserAuthSymmetricKey_Dev];
END;
GO
```

Procedure for Login

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROCEDURE [dbo].[USP_G_Login]
   @Email           NVARCHAR(MAX)=NULL,
   @Password        NVARCHAR(MAX)=NULL,
   @IsAuthenticated BIT=NULL OUTPUT,
   @Role            NVARCHAR(50)=NULL  OUTPUT,
   @UserName        NVARCHAR(500)=NULL OUTPUT,
   @ErrorMessage    NVARCHAR(MAX)=NULL OUTPUT
AS
BEGIN
   SET NOCOUNT ON;

   DECLARE @EncryptedPassword VARBINARY(MAX);
   DECLARE @DecryptedPassword NVARCHAR(MAX);

   IF NOT EXISTS (SELECT 1 FROM Users WHERE Email = @Email AND IsActive = 1)
   BEGIN
       SET @IsAuthenticated = 0;
       SET @ErrorMessage = 'Invalid Email Address';
       RETURN;
   END
```

```
    SELECT
        @EncryptedPassword = CAST(PasswordHash AS VARBINARY(MAX)),
        @UserName = UserName,
        @Role = (SELECT Name FROM Roles WHERE Id = U.RoleId)
    FROM Users U
    WHERE U.Email = @Email AND U.IsActive = 1;

    EXEC DecryptPassword @EncryptedPassword, @DecryptedPassword OUTPUT;

    IF @DecryptedPassword <> @Password
    BEGIN
        SET @IsAuthenticated = 0;
        SET @ErrorMessage = 'Invalid Password';
        RETURN;
    END

    SET @IsAuthenticated = 1;
    SET @ErrorMessage = NULL;

    SELECT
        U.Email,
        U.UserName,
        R.Name AS Role
    FROM Users U
    INNER JOIN Roles R ON U.RoleId = R.Id
    WHERE U.Email = @Email AND U.IsActive = 1;
END
GO
```

To check the actual Password

```
OPEN SYMMETRIC KEY UserAuthSymmetricKey_Dev
   DECRYPTION BY CERTIFICATE UserAuthCertificate_Dev;


SELECT
   Email,
   CONVERT(VARCHAR(MAX), DECRYPTBYKEY(PasswordHash)) AS DecryptedPassword
FROM
   Users;

CLOSE SYMMETRIC KEY UserAuthSymmetricKey_Dev;
```

Procedure for registering User:

```
```

```
ALTER PROCEDURE [dbo].[USP_Ins_RegisterAdminOrUser]
(
   @UserId   AS VARCHAR(100)=NULL,
   @Password AS VARCHAR(100)=NULL,
   @UserName AS VARCHAR(250)=NULL
)
AS
BEGIN
   SET NOCOUNT ON;

   DECLARE @EncryptedPassword VARBINARY(MAX);

   EXEC EncryptPassword @Password, @EncryptedPassword OUTPUT;

   BEGIN TRANSACTION;
   INSERT into Users
   (
       Email,
       PasswordHash,
       UserName,
       NormalizedEmail,
       NormalizedUserName,
       ConcurrencyStamp,
       RoleId,
       IsActive
   )
   VALUES
   (
       @UserId,
       @EncryptedPassword,
       @UserName,
       @UserId,
       @UserName,
       NEWID(),
       1,
       1
   )
   COMMIT TRANSACTION;

END;
GO
```

Table for creating Roles:

```
CREATE TABLE [dbo].[Roles](
   [Id] [bigint] IDENTITY(1,1) NOT NULL,
```

```
   [Name] [nvarchar](256) NOT NULL,
   [NormalizedName] [nvarchar](256) NOT NULL,
   [ConcurrencyStamp] [nvarchar](256) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Roles] ADD PRIMARY KEY CLUSTERED
(
   [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF,
IGNORE_DUP_KEY = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON
[PRIMARY]
GO
```

Table for Users:

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Users](
   [Id] [bigint] IDENTITY(1,1) NOT NULL,
   [UserName] [nvarchar](256) NOT NULL,
   [NormalizedUserName] [nvarchar](256) NOT NULL,
   [Email] [nvarchar](256) NOT NULL,
   [NormalizedEmail] [nvarchar](256) NOT NULL,
   [PasswordHash] [varchar](max) NOT NULL,
   [PasswordSalt] [nvarchar](max) NULL,
   [ConcurrencyStamp] [nvarchar](256) NULL,
   [RoleId] [int] NULL,
   [IsActive] [bit] NULL,
   [Otp] [varchar](10) NULL,
   [AccessToken] [varchar](1) NULL,
   [IsPassUpdated] [bit] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
ALTER TABLE [dbo].[Users] ADD PRIMARY KEY CLUSTERED
(
   [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF,
IGNORE_DUP_KEY = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON
[PRIMARY]
GO
ALTER TABLE [dbo].[Users] ADD  CONSTRAINT [DEFAULT_Users_IsActive]  DEFAULT ((1)) FOR
[IsActive]
GO
```