LANGUAGE LEARNING DATABASE MANAGEMENT SYSTEM

CONTENT

01 PURPOSE

02 DESIGN DOCUMENT

03 ENTITY RELATIONSHIP DIAGRAM

04 DATABASE IMPLEMENTATION

05 VISUALIZATIONS

06 CONLUSION

PURPOSE

Objective 1

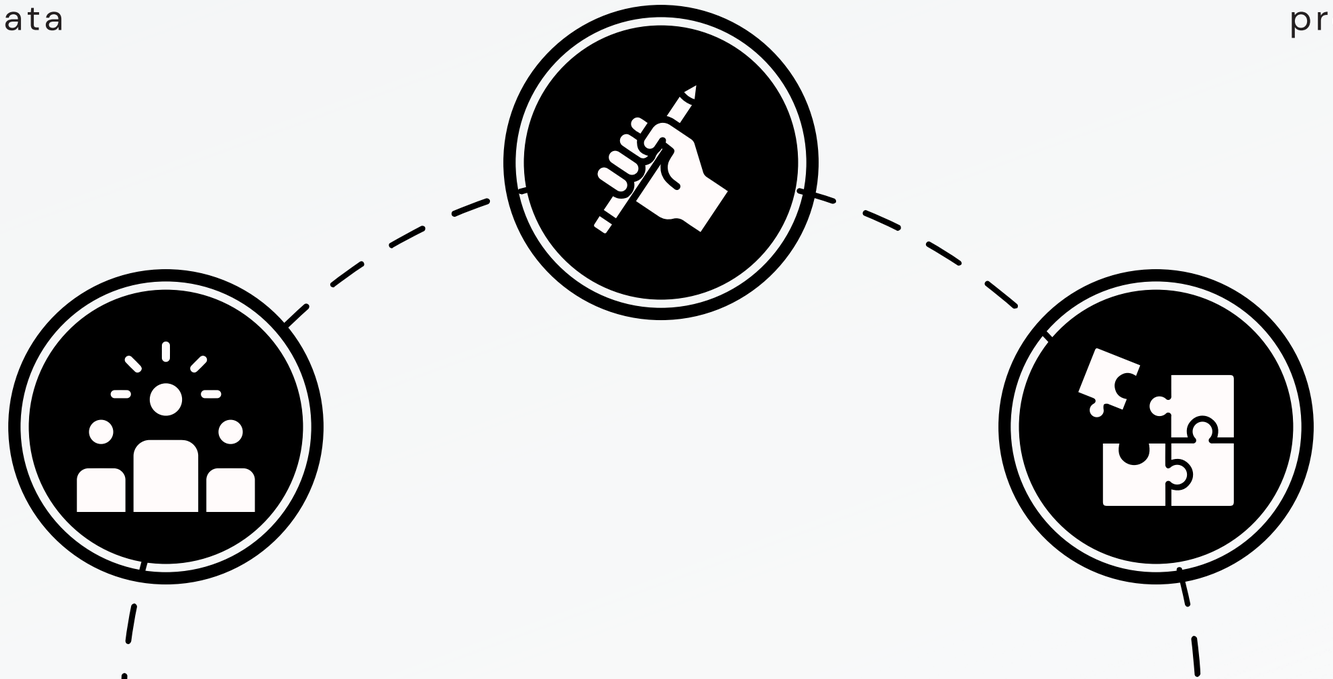
Reduce the cost of maintaining several disorganized systems by centralizing language learning data

Objective 2

Reduce the possibility of losing or duplicating data in language learning records

Objective 3

Encourage efficient, secure, and reliable language learning resource sharing and progress monitoring



DESIGN

01

REQUIREMENT ANALYSIS

Started analysis with problems, objectives. Analysising functional requirements like reporting, tracking.

02

DATABASE DESIGN

Designing database schema with entities , relationships.

03

SYSTEM FUNCTIONALITIES

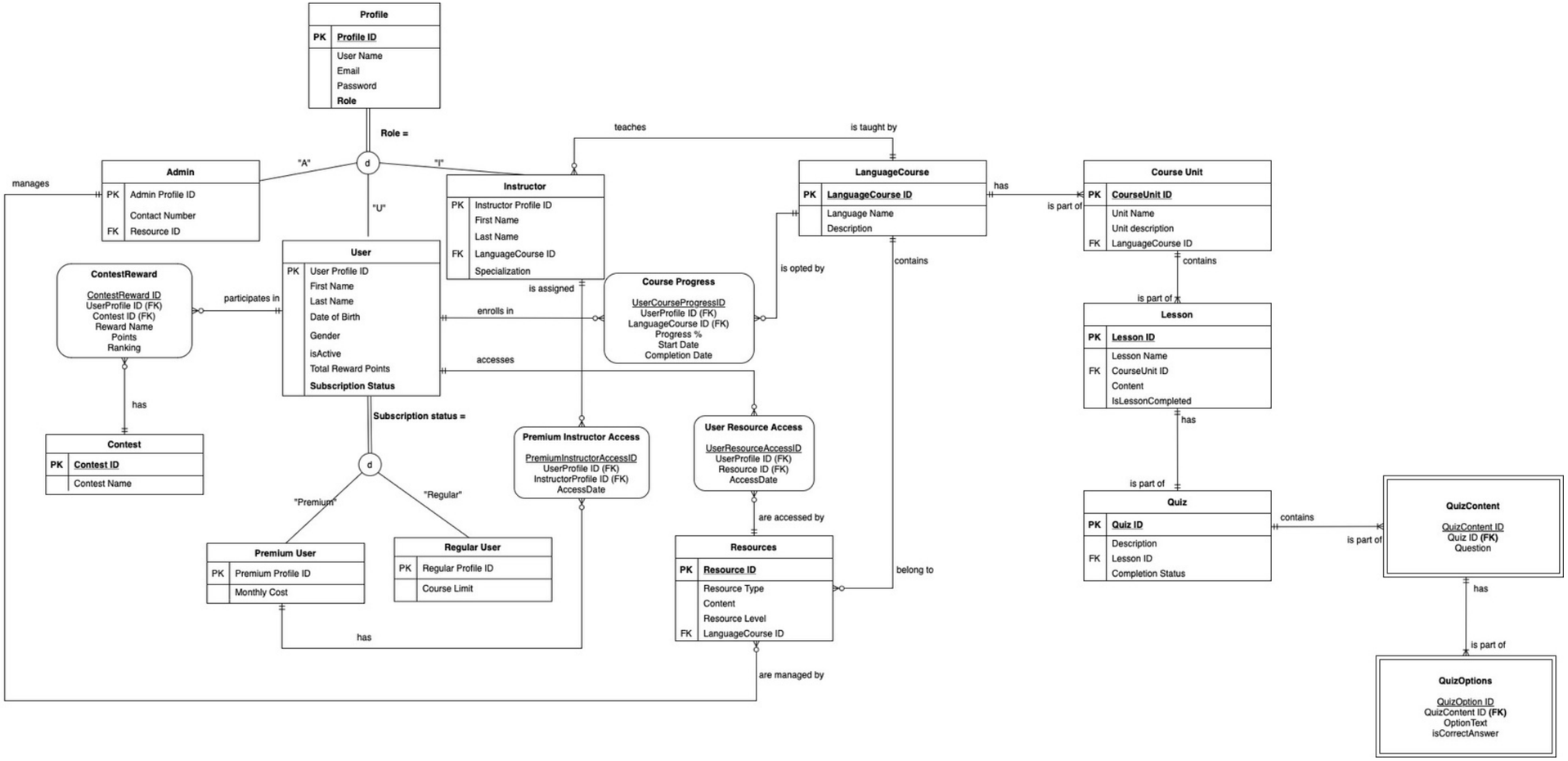
Started implementing several key functionalities to address business problems.

04

SECURITY MEASURES

Implements encryption for sensitive data such as passwords to ensure secure storage.

ER DIAGRAM


Profile
PK Profile ID
User Name
Email
Password
Role
Role =
d
"A"
"I"
"U"
Admin
PK Admin Profile ID
Contact Number
FK Resource ID
manages
Instructor
PK Instructor Profile ID
First Name
Last Name
FK LanguageCourse ID
Specialization
teaches
is taught by
LanguageCourse
PK LanguageCourse ID
Language Name
Description
has
is part of
Course Unit
PK CourseUnit ID
Unit Name
Unit description
FK LanguageCourse ID
contains
is part of
Lesson
PK Lesson ID
Lesson Name
FK CourseUnit ID
Content
IsLessonCompleted
has
is part of
Quiz
PK Quiz ID
Description
FK Lesson ID
Completion Status
contains
is part of
QuizContent
QuizContent ID
Quiz ID (FK)
Question
has
is part of
QuizOptions
QuizOption ID
QuizContent ID (FK)
OptionText
isCorrectAnswer
User
PK User Profile ID
First Name
Last Name
Date of Birth
Gender
isActive
Total Reward Points
Subscription Status
is assigned
enrolls in
accesses
ContestReward
ContestReward ID
UserProfile ID (FK)
Contest ID (FK)
Reward Name
Points
Ranking
participates in
has
Contest
PK Contest ID
Contest Name
Course Progress
UserCourseProgressID
UserProfile ID (FK)
LanguageCourse ID (FK)
Progress %
Start Date
Completion Date
is opted by
contains
Subscription status =
d
"Premium"
"Regular"
Premium User
PK Premium Profile ID
Monthly Cost
Regular User
PK Regular Profile ID
Course Limit
has
Premium Instructor Access
PremiumInstructorAccessID
UserProfile ID (FK)
InstructorProfile ID (FK)
AccessDate
User Resource Access
UserResourceAccessID
UserProfile ID (FK)
Resource ID (FK)
AccessDate
are accessed by
Resources
PK Resource ID
Resource Type
Content
Resource Level
FK LanguageCourse ID
belong to
are managed by

ENCRYPTION

```
---------------- COLUMN DATA ENCRYPTION ------------------------

CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'DMDDLLP@17';

SELECT name KeyName,
    symmetric_key_id KeyID,
    key_length KeyLength,
    algorithm_desc KeyAlgorithm
FROM sys.symmetric_keys;

CREATE CERTIFICATE MyLanguageLearningPlatformCertificate WITH SUBJECT = 'Password Column Encryption Certificate';

CREATE SYMMETRIC KEY MySymmetricKey WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE MyLanguageLearningPlatformCertificate;

-- Stored Procedure to encrypt the password while inserting to profile table and this procedure is called in the Trigger
CREATE PROCEDURE dbo.EncryptPasswordData
    @passwordInput NVARCHAR(255),
    @EncryptedData VARBINARY(8000) OUTPUT
AS
BEGIN
    SET NOCOUNT ON;
    OPEN SYMMETRIC KEY MySymmetricKey
    DECRYPTION BY CERTIFICATE MyLanguageLearningPlatformCertificate;
    SET @EncryptedData = ENCRYPTBYKEY(KEY_GUID('MySymmetricKey'), @passwordInput);
    CLOSE SYMMETRIC KEY MySymmetricKey;
END;

-- Decrypting the password

OPEN SYMMETRIC KEY MySymmetricKey
DECRYPTION BY CERTIFICATE MyLanguageLearningPlatformCertificate;

SELECT *, CONVERT(varchar, DecryptByKey([password])) AS DecryptedPassword FROM dbo.[profile];

CLOSE SYMMETRIC KEY MySymmetricKey;
```

```
CREATE TRIGGER trg_SaveUserToProfile
ON userProfile
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @EncryptedPassword VARBINARY(8000);

    -- Declare variables to hold values from inserted table
    DECLARE @Username NVARCHAR(255), @Email NVARCHAR(255), @Password NVARCHAR(2
    @Role NVARCHAR(50);

    -- Fetch values from the inserted table
    SELECT @Username = CONCAT(firstName, '_', lastName),
           @Email = email,
           @Password = CONCAT(lastName, '@' , userProfileId),
           @Role = 'User'
    FROM inserted;

    -- Call the EncryptPasswordData procedure to encrypt the password
    EXEC EncryptPasswordData @Password, @EncryptedPassword OUTPUT;

    -- Insert data into the profile table, including the encrypted password
    INSERT INTO profile (username, email, password, role)
    VALUES (@Username, @Email, @EncryptedPassword, @Role);
END;

    -- To verify the userProfile trigger
    Select * from profile
```

TRIGGERS

```
------------------------  TRIGGERS -------------------------------
-- Trigger for Auditing New User Registrations - 1
    CREATE TABLE Audit_UserRegistrations
    (
        auditId INT not null PRIMARY KEY IDENTITY(1,1),
        userProfileId INT not null,
        registrationDate DATETIME,
        userName NVARCHAR(255)
    );
    GO;

-- Creates a trigger to audit new user registrations on the 'userProfile' table.
    GO;
    CREATE TRIGGER trg_AuditNewUserRegistration ON userProfile
        AFTER
        INSERT AS
        BEGIN
        INSERT INTO Audit_UserRegistrations
            (userProfileId, registrationDate, userName)
        SELECT i.userProfileId, GETDATE(), i.firstName + ' ' + i.lastName
        FROM inserted i;
    END;

    -- TO execute if the trigger inserted into the table Audit_UserRegistrations
    Select *
    from Audit_UserRegistrations

-------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------

    -- To update reward points on userProfile - 2
    -- Creates a trigger to update the total reward points on the 'userProfile'
    -- table after contest rewards are inserted.
    GO;

    CREATE TRIGGER AfterContestRewardInsert ON contestReward
    AFTER INSERT
    AS
    BEGIN
    UPDATE userProfile
        SET totalRewardPoints = totalRewardPoints + inserted.points
        FROM inserted
        WHERE userProfile.userProfileId = inserted.userProfileId;
    END;

-------------------------------------------------------------------------
```

```
-------------------------------------------------------------------------

    -- Trigger 6: Insert User Subscription Data
    -- Creates a trigger to insert user subscription data into respective tables based on subscription status.

    GO;

    CREATE TRIGGER trg_InsertUserSubscription
    ON userProfile
    AFTER INSERT
    AS
    BEGIN
    INSERT INTO premiumUser
        (userProfileId)
    SELECT userProfileId
    FROM inserted
    WHERE subscriptionStatus = 'Premium';

    INSERT INTO regularUser
        (userProfileId)
    SELECT userProfileId
    FROM inserted
    WHERE subscriptionStatus = 'Regular';
    END;

    Select *
    from premiumUser
    Select *
    from regularUser
```

STORED PROCEDURES

```
------------------------  STORED PROCEDURES ---------------------

    -- USE LanguageLearningPlatform
    GO
-- Enroll in Language Course - 1

CREATE PROCEDURE EnrollInLanguageCourse
    @userProfileId INT,
    @languageCourseId INT,
    @startDate DATE,
    @enrollmentStatus BIT OUTPUT
AS
BEGIN-- Check if the user is already enrolled in the course
    IF NOT EXISTS (SELECT *
    FROM courseProgress
    WHERE userProfileId = @userProfileId AND languageCourseId = @languageCourseId)
        BEGIN
        INSERT INTO courseProgress
            (userProfileId, languageCourseId, progressPercent, startDate)
        VALUES
            (@userProfileId, @languageCourseId, 0, @startDate);
        SET @enrollmentStatus = 1;
    -- Successful enrollment
    END ELSE BEGIN
        SET @enrollmentStatus = 0;
    -- Enrollment failed, already enrolled
    END
END;
    GO

DECLARE @userProfileId INT = 69;
-- Specify the user profile ID
DECLARE @languageCourseId INT = 4;
-- Specify the language course ID
DECLARE @startDate DATE = '2024-03-16';
-- Specify the start date

DECLARE @enrollmentStatus BIT;
-- Output parameter

EXEC EnrollInLanguageCourse
        @userProfileId = @userProfileId,
        @languageCourseId = @languageCourseId,
        @startDate = @startDate,
        @enrollmentStatus = @enrollmentStatus OUTPUT;

SELECT @enrollmentStatus AS 'Enrollment Status';
```

```
-------------------------------------------------------------------------

    -- Award Contest Points - 2
    GO;

CREATE PROCEDURE AwardContestPoints
    @contestId INT,
    @userProfileId INT,
    @pointsAwarded INT,
    @ranking INT,
    @awardStatus BIT OUTPUT
AS
BEGIN-- Check if the awarding process is valid
    IF EXISTS (SELECT *
        FROM contest
        WHERE contestId = @contestId) AND EXISTS (SELECT *
        FROM userProfile
        WHERE userProfileId = @userProfileId)
        BEGIN
        INSERT INTO contestReward
            (contestId, userProfileId, points, ranking)
        VALUES
            (@contestId, @userProfileId, @pointsAwarded, @ranking);
        -- Update user's total points
        UPDATE userProfile
            SET totalRewardPoints = totalRewardPoints + @pointsAwarded WHERE userProfileId = @userProfileId;
        SET @awardStatus = 1;
    -- Successful awarding
    END ELSE BEGIN
        SET @awardStatus = 0;
    -- Awarding failed due to invalid contestId or userProfileId
    END
END;
    GO

-- Award Contest Points execution with below query

DECLARE @awardStatus BIT;

EXEC AwardContestPoints
        @contestId = 1,
        @userProfileId = 21,
        @pointsAwarded = 100,
        @ranking = 1,
        @awardStatus = @awardStatus OUTPUT;

SELECT @awardStatus AS AwardStatus;
```

UDF & INDEXES

```
================================================================
-----------------------  FUNCTIONS -----------------------------

-- UDF for Calculating Age - 1
CREATE FUNCTION dbo.fn_CalculateAge(@dateOfBirth DATE)RETURNS INT AS BEGIN
    DECLARE @today DATE = GETDATE();
    DECLARE @age INT;

    SET @age = DATEDIFF(YEAR, @dateOfBirth, @today) - CASE WHEN
MONTH
(@dateOfBirth) > MONTH
(@today) OR
        (MONTH
(@dateOfBirth) = MONTH
(@today) AND DAY
(@dateOfBirth) > DAY
(@today)) THEN 1 ELSE 0 END;

    RETURN @age;
END;

ALTER TABLE userProfile ADD computedAge AS dbo.fn_CalculateAge(dateOfBirth);

Select *
from userProfile

-------------------------------------------------------------------------
```

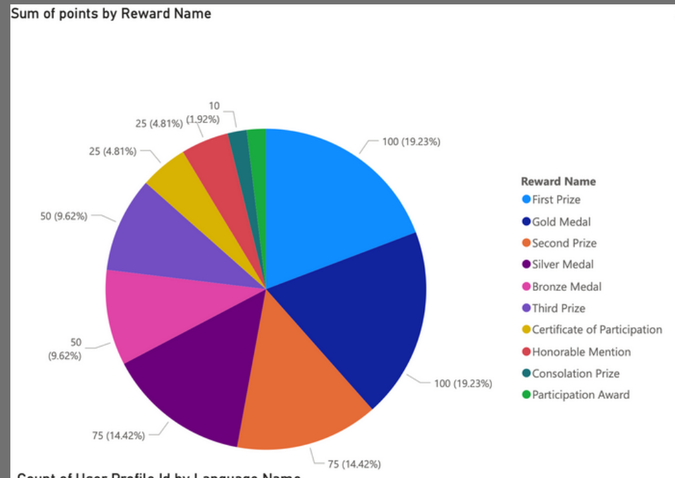
```
-- Index on userProfile for subscriptionStatus - 1

CREATE NONCLUSTERED INDEX idx_userProfile_subscriptionStatus ON userProfile (subscriptionStatus);

SELECT *
FROM sys.indexes
WHERE name = 'idx_userProfile_subscriptionStatus';

-- Index on languageCourse for languageName - 2

CREATE NONCLUSTERED INDEX idx_languageCourse_languageName ON languageCourse (languageName);

SELECT *
FROM sys.indexes
WHERE name = 'idx_languageCourse_languageName';

-- Index on resources for resourceLevel and languageCourseId - 3

CREATE NONCLUSTERED INDEX idx_resources_level_courseId ON resources (resourceLevel, languageCourseId);

SELECT *
FROM sys.indexes
WHERE name = 'idx_resources_level_courseId';

-- Index on courseProgress for progressPercent and userProfileId - 4

CREATE NONCLUSTERED INDEX idx_courseProgress_progress_userProfile ON courseProgress (progressPercent, userProfileId);

SELECT *
FROM sys.indexes
WHERE name = 'idx_courseProgress_progress_userProfile';
```

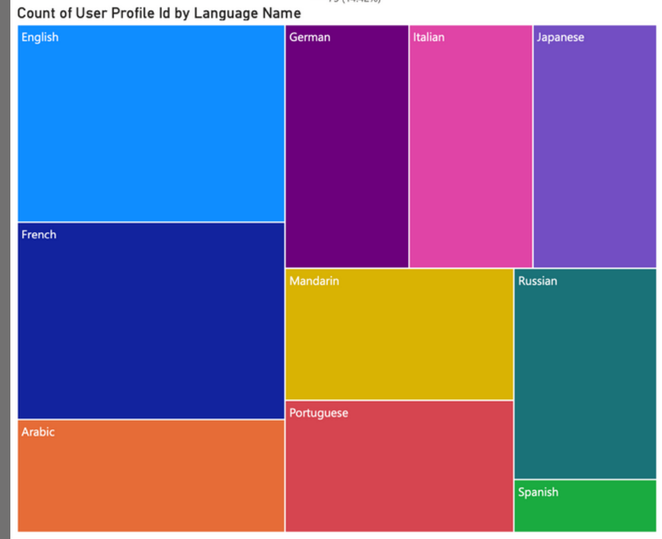
VIEWS

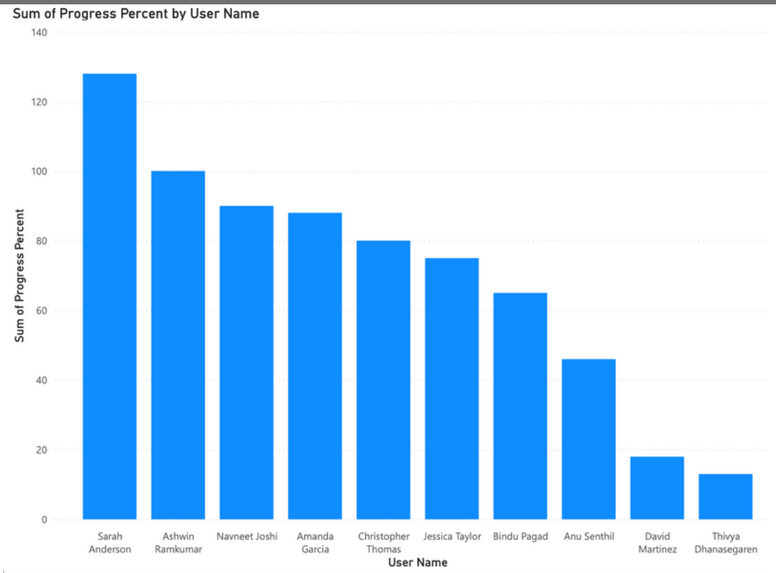
```
--------------------------  VIEWS ------------------------------

-- View for Instructor Course Details - 1

CREATE VIEW InstructorCourseDetails
AS
    SELECT ip.firstName, ip.lastName, ip.specialization, lc.languageName, lc.description
    FROM instructorProfile ip
        JOIN languageCourse lc ON ip.languageCourseId = lc.languageCourseId;

Select *
from InstructorCourseDetails

-------------------------------------------------------------------------

-- View for User Profiles with Premium Subscription Status - 2

CREATE VIEW PremiumUserProfiles
AS
    SELECT up.userProfileId, up.firstName, up.lastName, up.dateOfBirth, up.gender,
        up.totalRewardPoints
    FROM userProfile up
    WHERE up.subscriptionStatus = 'Premium';

Select *
from PremiumUserProfiles

-------------------------------------------------------------------------
```

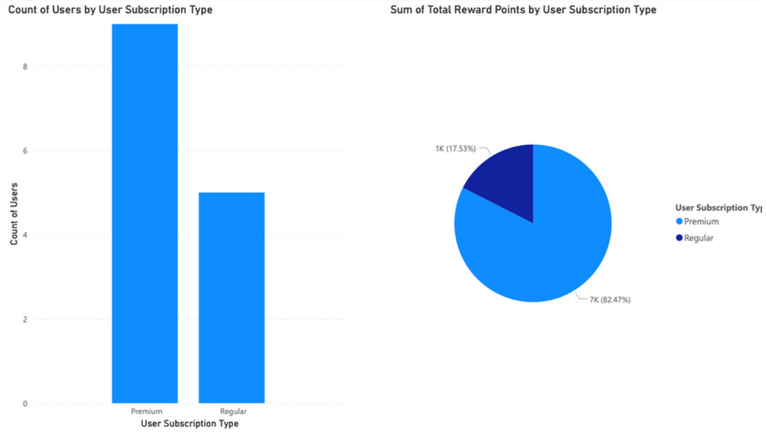
```
-------------------------------------------------------------------------

-- View for Course Progress Overview - 3
CREATE VIEW CourseProgressOverview
AS
    SELECT up.firstName, up.lastName, lc.languageName, cp.progressPercent, cp.startDate, cp.completionDate
    FROM courseProgress cp JOIN userProfile up ON cp.userProfileId = up.userProfileId
        JOIN languageCourse lc ON cp.languageCourseId = lc.languageCourseId;

Select *
from CourseProgressOverview
where progressPercent >= 50

-------------------------------------------------------------------------

-- View for Resource Access Logs - 4
CREATE VIEW UserResourceAccessLogs
AS
    SELECT up.firstName, up.lastName, r.resourceType, r.resourceContent, ura.accessDate
    FROM userResourceAccess ura JOIN userProfile up ON ura.userProfileId = up.userProfileId
        JOIN resources r ON ura.resourceId = r.resourceId;

Select *
from UserResourceAccessLogs
where resourceType = 'Book'

-------------------------------------------------------------------------

-- View for Contest Participation and Rewards - 5
CREATE VIEW ContestParticipationRewards
AS
    SELECT up.firstName, up.lastName, c.contestName, cr.rewardName, cr.points, cr.ranking
    FROM contestReward cr
        JOIN userProfile up ON cr.userProfileId = up.userProfileId
        JOIN contest c ON cr.contestId = c.contestId;

Select *
from ContestParticipationRewards
where ranking >= 1
```

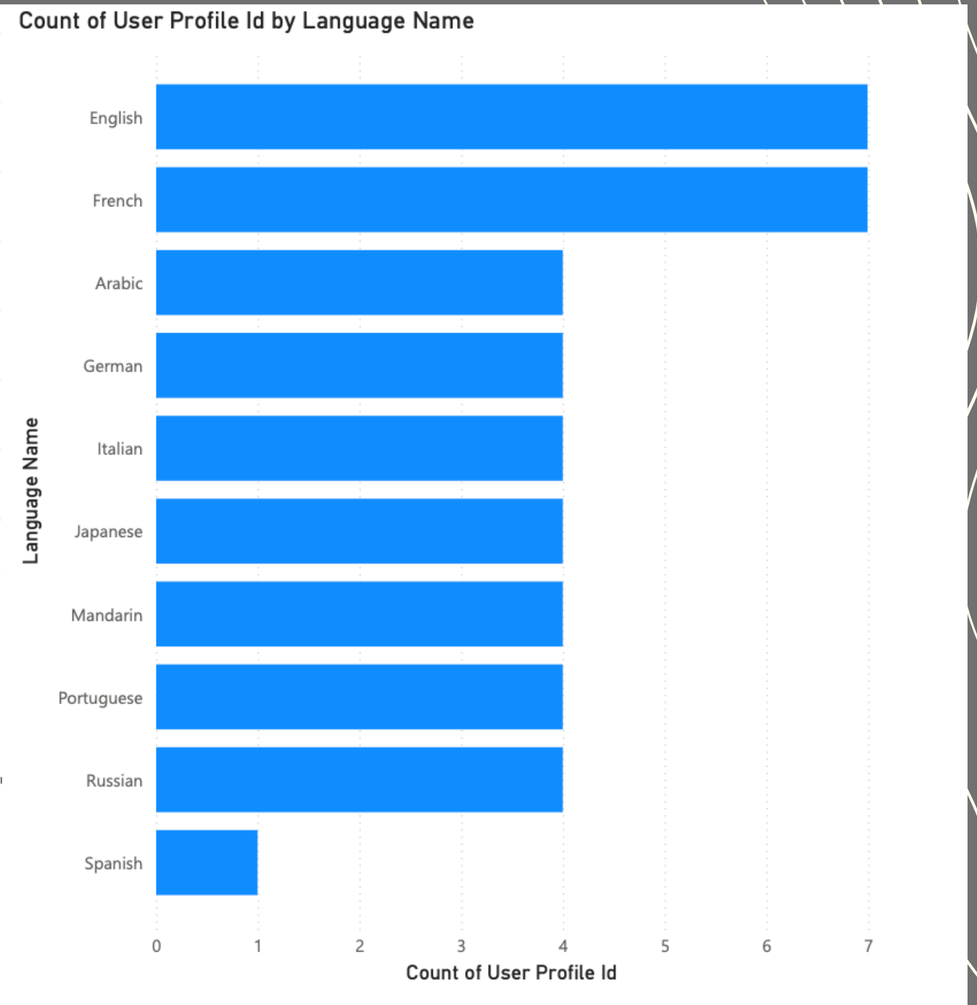
DATA VISUALIZATION









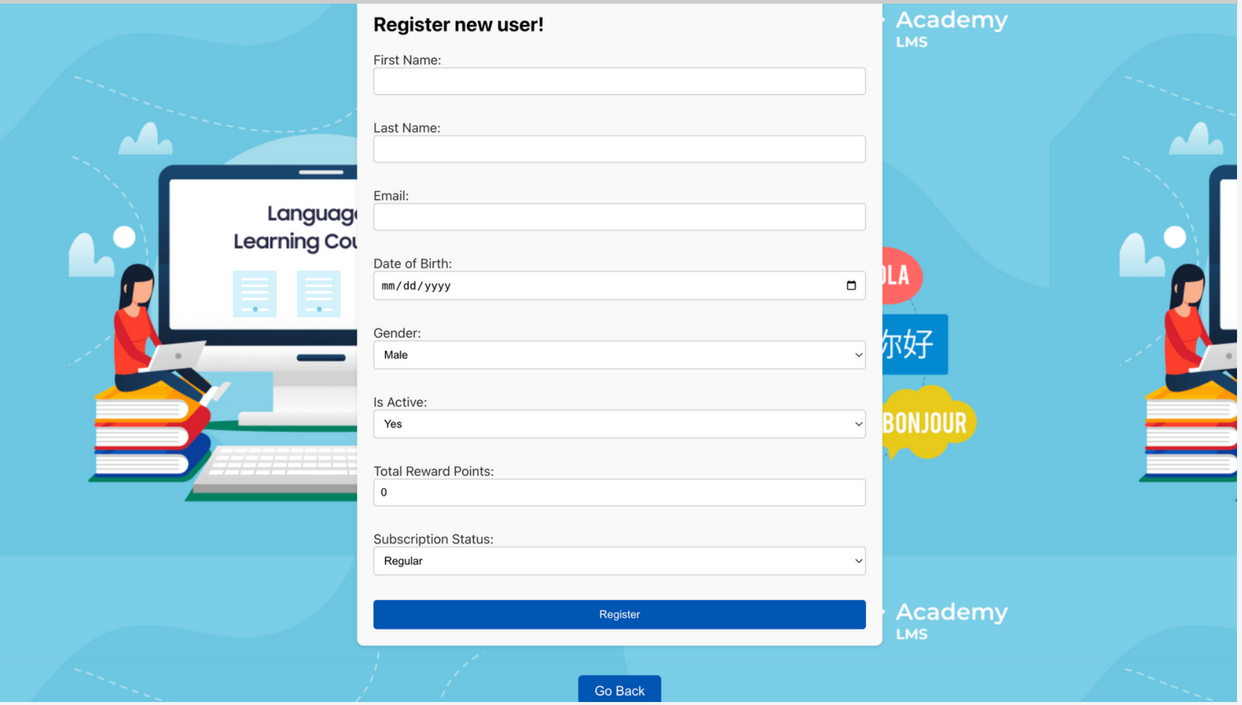


GUI

Application Home Page

GUI

New User Registration



User Operation

First Name	Last Name	Email	Actions
Bindu	Pagad	bindu.pagad@llp.com	Update
Anu	Senthil	anu.senthil@llp.com	Update
Thivya	Dhanasegaren	thivya.dhanasegaren@llp.com	Update
Ashwin	Ramkumar	ashwin.ramkumar@llp.com	Update
Navneet	Joshi	navneet.joshi@llp.com	Update
Sarah	Anderson	sarah.anderson@llp.com	Update
David	Martinez	david.martinez@llp.com	Update
Jessica	Taylor	jessica.taylor@llp.com	Update
Christopher	Thomas	christopher.thomas@llp.com	Update
Amanda	Garcia	amanda.garcia@llp.com	Update
Alex	Jones	alex123@llp.com	Update
Neymar	Jr	neymar@llp.com	Update
ronaldinho	gaucho	ronaldinho@gmail.com	Update
Chimy	Avila	chimyavila@gmail.com	Update

GROUP 17

Thivya

Bindu

Anusha

Navneet

Aswin

THANK YOU