1. Project Overview

TEDUWallet (also referred to as CampusCoin) is a campus-based digital economy system designed to simulate real-world financial interactions. The system allows university students to earn "CampusCoins" by completing administrative or social tasks and spend them on rewards like cafeteria discounts or stationery coupons.

2. Actors

- **Student:** Participants who perform tasks to earn coins and purchase rewards.
- **Admin (University Rectorate):** Managers who create tasks, validate completion, and monitor system statistics.
- **Vendor (Partner/Sponsor):** External or internal entities providing the rewards or discounts.

3. User Stories

As a Student:

- **Registration/Login:** I want to log in using my university credentials to access my personal wallet.
- **Task Discovery:** I want to browse available tasks and filter them by eligibility (e.g., department, class level) to find opportunities to earn coins.
- **Task Application:** I want to apply for tasks directly through the system so I can participate before the quota fills.
- **Wallet Management:** I want to view my current coin balance and transaction history to track my earnings and spending.
- **Reward Redemption:** I want to use my balance to purchase coupons or discounts from the Reward Market.

As an Admin:

- **Task Management:** I want to create, edit, and delete tasks with details like deadlines, quotas, and reward amounts.
- **Completion Approval:** I want to verify and approve student task completion to trigger the automatic transfer of reward points.
- **Monitoring:** I want to view a dashboard with statistics on active tasks, student participation, and total distributed points.
- **Notification:** I want to send announcements about new tasks to students.

Use Case ID	Use Case Name	Actor	Description
UC-01	**Manage Tasks**	Admin	Admin defines task parameters (description, date, reward, quota) and publishes them to the system[15].
UC-02	**Apply for Activity**	Student	Student views task details and submits an application; system checks if the quota is full[16].
UC-03	**Verify Completion**	Admin	Admin reviews the student's work and changes the status to "Completed," triggering the reward transaction[171717].
UC-04	**Redeem Reward**	Student	Student selects a reward; the system verifies sufficient balance, deducts coins, and generates a unique coupon code[181818181818].
UC-05	**View Audit Logs**	Admin	Admin views a history of all system actions (applications, spending, status changes) for security and transparency[19191919].

5. Scenario Flows

Flow A: The Earning Cycle (Task Completion)

1. **Creation:** The Admin creates a task (e.g., "Career Fair Assistant") with a specific coin reward.
2. **Application:** The Student browses the task list, checks eligibility, and clicks "Apply".
3. **Execution:** The Student performs the required duty.
4. **Approval:** The Admin marks the task as "Completed" in the system.
5. **Payment:** The system triggers `UpdateWalletOnCompletion`, automatically adding the specific amount of coins to the Student's wallet.

Flow B: The Spending Cycle (Reward Redemption)

1. **Selection:** The Student visits the "Reward Market" and selects an item (e.g., "Coffee Voucher").
2. **Validation:** The system checks if the Student's wallet balance ≥ Reward Cost.
3. **Transaction:** If valid, the coin amount is deducted from the wallet immediately.
4. **Logging:** The transaction is recorded in the `WALLET_SPENDS_REWARD` table and the main `LOG` table for auditing.

6. IMPORTANT RULES

This database uses DB-First approach. While generating any code, do not override this setting. Always use DB-first approach.

7. FULL DATABASE SQL SCHEMA

```
CREATE DATABASE [CAMPUSCOIN];
GO
USE [CAMPUSCOIN];
GO

/* ================ TABLES ================ */

CREATE TABLE [dbo].[ADMIN](
   [AdminId] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
   [Name] [varchar](30) NOT NULL,
   [Surname] [varchar](30) NULL,
   [Position] [varchar](20) NOT NULL,
   [Password] [varchar](255) NULL,
   [Username] [varchar](20) NOT NULL,
   [email] [varchar](50) NULL
);

CREATE TABLE [dbo].[STUDENT](
```

```
    [StudentId] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
    [Name] [varchar](30) NOT NULL,
    [Surname] [varchar](30) NULL,
    [email] [varchar](50) NULL,
    [Departmant] [varchar](20) NOT NULL,
    [Password] [varchar](255) NULL,
    [Username] [varchar](20) NOT NULL,
    [Coins] [decimal](18, 0) NULL
);

CREATE TABLE [dbo].[ACTIVITY](
    [ActivityId] [int] NOT NULL PRIMARY KEY, -- Note: Not set as Identity in original
    [Title] [varchar](30) NOT NULL,
    [Description] [varchar](255) NULL,
    [Status] [varchar](30) NULL,
    [RewardTokenAmount] [decimal](18, 0) NOT NULL,
    [MaxParticipants] [int] NOT NULL,
    [CreatedDate] [date] NULL,
    [AdminId] [int] NOT NULL,
    [Deadline] [date] NULL,
    [PriorityLevel] [decimal](18, 0) NULL,
    CONSTRAINT [FK_Activity_Admin] FOREIGN KEY([AdminId]) REFERENCES [dbo].[ADMIN]
([AdminId])
);

CREATE TABLE [dbo].[REWARD](
    [RewardId] [int] NOT NULL PRIMARY KEY,
    [RewardName] [varchar](100) NOT NULL,
    [RewardType] [varchar](50) NOT NULL,
    [Cost] [decimal](18, 0) NOT NULL,
    [CreatedDate] [date] NULL,
    [Vendor] [varchar](100) NULL,
    [UniqueCode] [varchar](50) NOT NULL,
    [Status] [varchar](20) NOT NULL,
    [ExpiryDate] [date] NULL,
    CONSTRAINT [UQ_Reward_UniqueCode] UNIQUE ([UniqueCode])
);

CREATE TABLE [dbo].[WALLET](
    [WalletId] [int] NOT NULL PRIMARY KEY,
    [Balance] [decimal](18, 0) NOT NULL,
    [StudentId] [int] NOT NULL,
    [LastUpdated] [date] NULL,
    [CreatedDate] [date] NULL,
```

```
    CONSTRAINT [FK_Wallet_Student] FOREIGN KEY([StudentId]) REFERENCES
[dbo].[STUDENT] ([StudentId]),
    CONSTRAINT [UQ_Wallet_StudentId] UNIQUE ([StudentId])
);

CREATE TABLE [dbo].[APPLY](
    [StudentId] [int] NOT NULL,
    [ActivityId] [int] NOT NULL,
    [ApplicationDate] [date] NOT NULL,
    [Status] [varchar](20) NOT NULL,
    CONSTRAINT [PK_Apply] PRIMARY KEY ([StudentId], [ActivityId]),
    CONSTRAINT [FK_Apply_Student] FOREIGN KEY([StudentId]) REFERENCES
[dbo].[STUDENT] ([StudentId]),
    CONSTRAINT [FK_Apply_Activity] FOREIGN KEY([ActivityId]) REFERENCES
[dbo].[ACTIVITY] ([ActivityId])
);

CREATE TABLE [dbo].[COMPLETES](
    [StudentId] [int] NOT NULL,
    [ActivityId] [int] NOT NULL,
    [CompletionDate] [date] NOT NULL,
    [AwardedAmount] [decimal](18, 0) NOT NULL,
    CONSTRAINT [PK_Completes] PRIMARY KEY ([StudentId], [ActivityId]),
    CONSTRAINT [FK_Completes_Student] FOREIGN KEY([StudentId]) REFERENCES
[dbo].[STUDENT] ([StudentId]),
    CONSTRAINT [FK_Completes_Activity] FOREIGN KEY([ActivityId]) REFERENCES
[dbo].[ACTIVITY] ([ActivityId])
);

CREATE TABLE [dbo].[LOG](
    [LogId] [int] IDENTITY(1,1) NOT NULL PRIMARY KEY,
    [StudentId] [int] NOT NULL,
    [ActivityId] [int] NULL,
    [RewardId] [int] NULL,
    [ActionType] [varchar](50) NOT NULL,
    [Timestamp] [datetime2](7) NULL,
    CONSTRAINT [FK_Log_Student] FOREIGN KEY([StudentId]) REFERENCES
[dbo].[STUDENT] ([StudentId]),
    CONSTRAINT [FK_Log_Activity] FOREIGN KEY([ActivityId]) REFERENCES
[dbo].[ACTIVITY] ([ActivityId]),
    CONSTRAINT [FK_Log_Reward] FOREIGN KEY([RewardId]) REFERENCES
[dbo].[REWARD] ([RewardId])
);
```

```
CREATE TABLE [dbo].[WALLET_SPENDS_REWARD](
   [TransactionId] [int] NOT NULL PRIMARY KEY,
   [WalletId] [int] NOT NULL,
   [RewardId] [int] NOT NULL,
   [SpentDate] [date] NOT NULL,
   [UsedDate] [date] NOT NULL,
   [Quantity] [int] NOT NULL,
   [StudentId] [int] NULL,
   CONSTRAINT [FK_WSR_Wallet] FOREIGN KEY([WalletId]) REFERENCES [dbo].[WALLET]
([WalletId]),
   CONSTRAINT [FK_WSR_Reward] FOREIGN KEY([RewardId]) REFERENCES
[dbo].[REWARD] ([RewardId]),
   CONSTRAINT [FK_WSR_Student] FOREIGN KEY([StudentId]) REFERENCES
[dbo].[STUDENT] ([StudentId])
);
GO

/* ================= INDEXES ================= */

CREATE NONCLUSTERED INDEX [idx_student_email] ON [dbo].[STUDENT]([email] ASC);
CREATE NONCLUSTERED INDEX [idx_student_username] ON [dbo].[STUDENT]([Username]
ASC);
CREATE NONCLUSTERED INDEX [idx_Log_Timestamp] ON [dbo].[LOG]([Timestamp] DESC);
GO

/* ================= VIEWS ================= */

CREATE VIEW [dbo].[vw_Top3StudentsThisWeek] AS
SELECT
   S.StudentId,
   S.Name + ' ' + S.Surname AS FullName,
   SUM(A.RewardTokenAmount) AS WeeklyCoinsEarned
FROM STUDENT S
INNER JOIN COMPLETES C ON S.StudentId = C.StudentId
INNER JOIN ACTIVITY A ON C.ActivityId = A.ActivityId
WHERE
   C.CompletionDate >= DATEADD(week, DATEDIFF(week, 0, GETDATE()), 0)
   AND C.CompletionDate < DATEADD(week, DATEDIFF(week, 0, GETDATE()) + 1, 0)
GROUP BY S.StudentId, S.Name, S.Surname;
GO

CREATE VIEW [dbo].[vw_student_completions] AS
SELECT c.StudentId, c.ActivityId, c.CompletionDate, c.AwardedAmount FROM COMPLETES c;
GO
```

```
CREATE VIEW [dbo].[vw_activity_details] AS
SELECT
   a.ActivityId, a.Title, a.Description, a.Status, a.RewardTokenAmount,
   a.MaxParticipants, a.Deadline, ad.Name AS AdminName, ad.Surname AS AdminSurname
FROM ACTIVITY a JOIN ADMIN ad ON a.AdminId = ad.AdminId;
GO

CREATE VIEW [dbo].[vw_student_wallet] AS
SELECT s.StudentId, s.Name, s.Surname, s.Username, w.WalletId, w.Balance, w.LastUpdated
FROM STUDENT s INNER JOIN WALLET w ON s.StudentId = w.StudentId;
GO

/* ================ TRIGGERS ================ */

CREATE TRIGGER [dbo].[logApplyActivity] ON [dbo].[APPLY]
AFTER INSERT AS BEGIN
   SET NOCOUNT ON;
   INSERT INTO LOG (ActionType, StudentId, ActivityId, [Timestamp])
   SELECT 'APPLY', i.StudentId, i.ActivityId, SYSUTCDATETIME() FROM inserted i;
END;
GO

CREATE TRIGGER [dbo].[logCompletesActivity] ON [dbo].[COMPLETES]
AFTER INSERT AS BEGIN
   SET NOCOUNT ON;
   INSERT INTO LOG (ActionType, StudentId, ActivityId, [Timestamp])
   SELECT 'COMPLETED', i.StudentId, i.ActivityId, SYSUTCDATETIME() FROM inserted i;
END;
GO

CREATE TRIGGER [dbo].[trg_UpdateWalletOnCompletion] ON [dbo].[COMPLETES]
AFTER INSERT AS BEGIN
   UPDATE W
   SET Balance = W.Balance + A.RewardTokenAmount
   FROM WALLET W
   INNER JOIN INSERTED I ON W.StudentId = I.StudentId
   INNER JOIN ACTIVITY A ON I.ActivityId = A.ActivityId
END;
GO

CREATE TRIGGER [dbo].[logSpends] ON [dbo].[WALLET_SPENDS_REWARD]
AFTER INSERT AS BEGIN
   SET NOCOUNT ON;
```

```
    INSERT INTO dbo.LOG (ActionType, StudentId, RewardId, [Timestamp])
    SELECT 'REWARD_SPENT', i.StudentId, i.RewardId, SYSUTCDATETIME() FROM inserted
i;
END;
GO
```