Specification

Introduction

The aim of this document is to outline the specifications for a mobile cross-platform application that enhances user engagement through interactive features. The application is designed to be user-friendly, secure, and accessible in multiple languages.

Application Overview

Imagine an app that motivates you to be and do good! Every day, you'll receive a personalised task that benefits either yourself or the wider community. Completing these tasks earns you virtual coins. Here's the twist: these coins can be converted into real money donated to a charity you choose. The app empowers you to make a positive difference, big or small, while rewarding your good deeds. By turning good intentions into tangible impact, this app creates a win-win for you, society, and your favourite cause. And on top of that, witnessing your positive actions accumulate translates into a powerful boost for your self-esteem.

Application Components

- Accounts (Users & Referrals)
- Items
- Charities
- Coins
- Events (Tasks & Habits)
- Settings

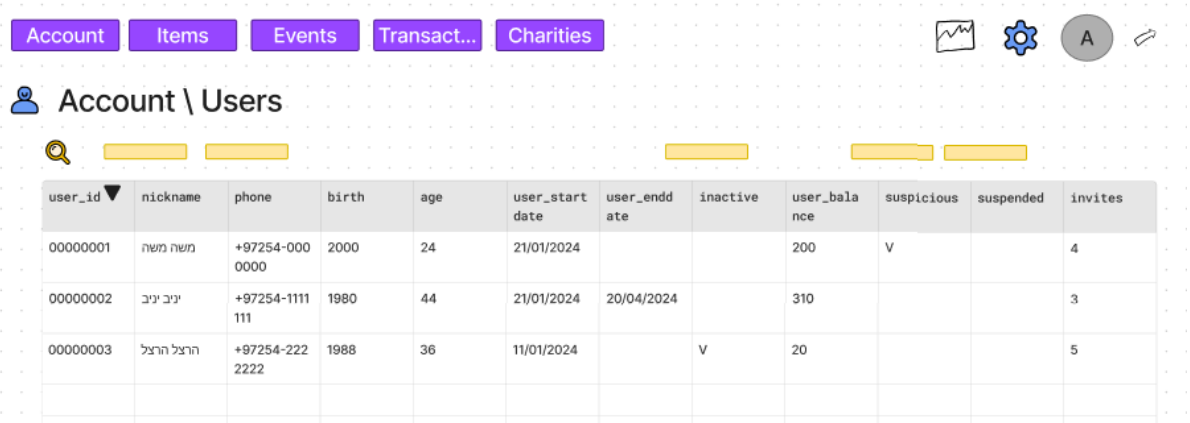
Dashboard admin

Overview

- Admin Dashboard will manage the following entities of the application:
    - Users & Referrals
    - Items
    - Events
    - Charities
    - Transactions
- Admin Dashboard should be built as a WEB application and it will be protected by username & password

Page Layout

- Top Menu for navigation between Components.
- Navigating inside the Component between the entities.
    - Account > 1. Users 2. Referrals
    - Items > 1. Items 2. Ignore List
    - Events > 1. Habits 2. Tasks
    - Transactions > 1. TR_ITEM 2. TR_OTHER
    - Charities > Charities
- In addition, in the top right corner admin will have access to the following:
    - System settings & Reference DB tables
    - App dashboard and statistics
    - Admin user actions



user_id	nickname	phone	birth	age	user_start date	user_endd ate	inactive	user_bala nce	suspicious	suspended	invites
00000001	משה משה	+97254-000 0000	2000	24	21/01/2024			200	V		4
00000002	יניב יניב	+97254-1111 111	1980	44	21/01/2024	20/04/2024		310			3
00000003	הרצל הרצל	+97254-222 2222	1988	36	11/01/2024		V	20			5

Accounts

1. A screen that displays all the users that signed up to the application.
2. It will be managed in a table with Order and Search capabilities (will be detailed below)
3. Accounts entity will include 2 tables. AC_USERS and AC_REFERRALS
4. User_id = 1 is the Admin user of the system

Accounts > AC_USERS

1. User fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
user_id	ID	INT	X	AUTO_INCREMENT
nickname	Account Name	VARCHAR(100)	Y	Sign up
phone	Phone number	INT	X	Sign up
birth	Year of birth	INT	Y	Sign up
age	Age	INT	Y	Calculation: Current year - Year of birth
user_startdate	Joined	DATE	Y (Rejoining)	Sign up date
user_enddate	Left	DATE	Y (Account deletion)	Account deletion date
inactive	User activity	BOOLEAN	Y	Calculation: User doesn't have 1 task or habit completion in the past **14** days (Will be taken from Settings Inactive_days)
user_incoins	In	INT	Y	Feed from Transactions tables
user_outcoins	Out	INT	Y	Feed from Transactions tables

user_balance	Balance	INT	Y	= user_incoins - user_outcoins
suspicious	Check user activities	BOOLEAN	Y	Events validations
suspended	User is suspended	BOOLEAN	Y	If the user is suspended, the user is blocked and cannot perform any action in the app.
Invites	Invites	INT	Y	Calculation: 5 (can be defined by settings) - user follows

2. Actions available in the table: Edit (only editable fields)
3. Table options:
    a. Sort by
        i. user_id (default)
        ii. start_date
    b. Filter/Search by
        i. nickname
        ii. phone
        iii. active
        iv. suspicious
        v. suspended

Accounts > AC_REFERRALS

1. In order to support the pyramid model, it's needed to add one more table to maintain the relationships between users (Lead, Follow)
2. A lead will enjoy a one time bonus of a successful invitation (another user signed up following his URL) = 20 coins
3. Maximum follows allowed - 5 signed up users (100 coins MAX forever)
4. Referral fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
referral_id	ID	INT (Unique, Primary key)	X	AUTO_INCREMENT
referrer_id	Lead username	INT (Foreign Key)	X	T_USERS
nickname	Lead Name	VARCHAR(100)	X	Sign up

referred_id	Follow username	INT (Foreign Key)	X	T_USERS
nickname	Follow Name	VARCHAR(100)	X	Sign up
referral_startdate	Created	DATE	X	Follow Sign up date
referral_enddate	Discontinued	DATE	X	User deletion date
rj_transaction	Joining bonus	INT	X	20 Joining bonus to referrer

5. Supporting user deletion scenarios by inserting End date and clearing it while rejoining after deletion.
    - Follow deletion
    - Follow rejoined
    - Lead deletion
    - Lead rejoined
6. Actions available in the table: N/A
7. Table options:
    a. Sort by
        i. referral_id (default)
        ii. referrer_id
        iii. referred_id
    b. Filter/Search by
        i. referrer_id
        ii. referred_id

Items

1. Items in fact are the core component of this application.
2. Item is what the user is committing to do everyday, in order to earn coins.
3. Items will be differentiated by difficulty level: Silver, Gold and Diamond.
4. Accomplishing items worth coins to the user as the following: (TD_Itemlevel)
    a. Silver - 1 coin | As for a referral - None
    b. Gold - 5 coins | As for a referral - 1 coin
    c. Diamond - 10 coins | As for a referral - 2 coins
5. In the Items component we will manage 2 tables: IT_ITEMS, IT_IGNORE.

Items > IT_ITEMS

1. Item fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
item_id	ID	INT	X	AUTO_INCREMENT
item_summary	Summary	VARCHAR(100)	Y	Text
item_type	Item Type	VARCHAR(20)	Y	TD_Itemtype
item_level	Level	VARCHAR(20)	V	TD_Itemlevel
item_body	Action	VARCHAR(100)	Y	TEXT
item_selected	Selected	INT	N	Count: Number of times task was selected by users
item_completed	Accomplished	INT	N	Count: Number of times task was marked as Completed by users
item_engagement	Engagement	INT %	N	Calculation: completed/selected
item_ignored	Ignore	INT	N	Count: Number of times task was marked as Ignore by users

2. Actions available in the table: Add,Edit (only editable fields),Delete, View

3. Table options:
    a. Sort by
        i. item_id (default)
        ii. item_engagement
        iii. item_ignore
    b. Filter/Search by
        i. item_summary
        ii. item_type
        iii. item_level

Items > IT_IGNORE

1. The application will allow users to mark items (Habit or Task) as ‘Ignore’ so they will not show up again later.
2. Ignore list will be accessible for the user, so they will be able to unignore items.
3. Item fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
ignore_id	ID	INT	X	AUTO_INCREMENT
user_id	User ID	INT	X	AC_Users.user_id
item_type	Item Type	VARCHAR(20)	X	TD_Itemtype
ignore_startdate	Ignoring Day	DATE	X	Ignoring date
ignore_enddate	Unignoring Date	DATE	X	Unignoring date

4. Actions available in the table: N/A
5. Table options:
    a. Sort by
        i. ignore_id (default)
    b. Filter/Search by
        i. user_id
        ii. item_type

Events

1. Events are the actions that the user triggered by his app:
    a. Selecting an item > Select
    b. Completing an item > Done
2. Event must be created because 2 main reasons:
    a. Indicate suspicious user for a fraud (Selecting an item and complete it immediately)
    b. Due to the items limitations (check TD_itemlevel.limit)
3. Aborting a selected item will add an end date to the event.
4. Events records will be divided to 2 separate tables: EV_Tasks and EV_Habits based on the item_type (Items.item_type)

Events > EV_HABITS

1. Habit fields to be displayed in the table: (AC_Users.user_id = AC_REFERRALS.referred_id)

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
habit_id	ID	INT	X	AUTO_INCREMENT
event_type	Event Type	VARCHAR(20)	X	TD_Eventtype
user_id	User ID	INT	X	AC_Users.user_id
nickname	Account Name	VARCHAR(100)	X	AC_Users.nickname
item_id	Item ID	INT	X	Items.item_id
habit_created	Created	DATE	X	Created Date
habit_enddate	End Date	DATE	X	Select - Abort Done - N/A
habit_sum	Done	INT	X	DISTINCT Habit Done user_id, item_id, Created > 21
habit_left	Days to completion	INT	X	= Calculation: 21 - event_habitsum
referrer_id	Lead User	INT	X	AC_REFERRALS.referrer_id Fill only if Event

				type is “Done”
event_ic	Coins Earned	INT	X	Calculation based on rules in: TD_Itemlevel Fill only if Event type is “Done”
event_rc	Coins for Lead	INT	X	Calculation based on rules in: TD_Itemlevel Fill only if Event type is “Done” And referrer_id is not empty

2. Actions available in the table: None
3. Table options:
    a. Sort by
        i. habit_id (default)
        ii. habit_created
        iii. event_type
    b. Filter/Search by
        i. user_id
        ii. item_id
        iii. event_type

Events > EV_TASKS

1. Task fields to be displayed in the table (AC_Users.user_id = AC_REFERRALS.referred_id):

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
task_id	ID	INT	X	AUTO_INCREMENT
event_type	Event Type	VARCHAR(20)	X	TD_Eventtype
user_id	User ID	INT	X	AC_Users.user_id
nickname	Account Name	VARCHAR(100)	X	AC_Users.nickname
item_id	Item ID	INT	X	Items.item_id
task_created	Created	DATE	X	Created Date
task_enddate	End Date	DATE	X	Select - Abort Done - N/A

referrer_id	Lead User	INT	X	AC_REFERRALS.referrer_id Fill only if Event type is “Done”
event_ic	Coins Earned	INT	X	Calculation based on rules in: TD_Itemlevel Fill only if Event type is “Done”
event_rc	Coins for Lead	INT	X	Calculation based on rules in: TD_Itemlevel Fill only if Event type is “Done” And referrer_id is not empty

2. Actions available in the table: None
3. Table options:
    a. Sort by
        i. task_id (default)
        ii. task_created
        iii. event_type
    b. Filter/Search by
        i. user_id
        ii. item_id
        iii. event_type

Transactions

1. Transactions are the coin movements in the application.
2. There are 5 kinds of movements: (Will be managed under TD_Transactiontype
    a. Referral “Join” - A one time 20 coins bonus, up to 5 life time.
    b. Referred item completion - Based on TD_Itemlevel.referral values.
    c. Item completion - Based on TD_Itemlevel.coins values.
    d. Donation - Transferring coins from User to Charity.
    e. Exchange - Exchange virtually coins to real money under Charity Component.
3. In the Transactions component we will manage 2 tables: TR_ITEM, TR_OTHER.

Transactions > TR_ITEM

1. Once the user is creating a Done event, a transaction will be created. If the user_id has referrer_id then 2 transactions will be created. One for the user himself and 2nd for his referrer. Following values in TD_Itemlevel, referral column.
2. Item fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
Transactionitem_id	ID	INT	X	AUTO_INCREMENT
Transactionitem_created	Created	DATE	X	Creation date
transaction_type	Transaction Type	VARCHAR(20)	X	TD_Transaction type
task_id	ID	INT	X	EV_TASKS.task_id
habit_id	ID	INT	X	EV_HABITS.habit_id
user_id	User ID	INT	X	AC_Users.user_id The receiver
transactionitem_amount	Amount	INT	X	Value is taken from: event_rc or event_ic

3. Actions available in the table: N/A
4. Table options:
    a. Sort by
        i. Transactionitem_id (default)
        ii. transaction_type
    b. Filter/Search by
        i. user_id

ii. task_id
iii. Habit_id
iv. transaction_type

Transactions > TR_OTHER

1. There are 3 kinds of transactions which are not related to events in the system:
    a. Referral program (User joined)
    b. Donate (transaction from User to Charity)
    c. Exchange (transaction from Charity to real money)
2. Item fields to be displayed in the table:

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
Transactionother_id	ID	INT	X	AUTO_INCREMENT
Transactionother_created	Created	DATE	X	Creation date
transaction_type	Transaction Type	VARCHAR(20)	X	TD_Transaction type
beneficial_type	Beneficial Type	INT	X	TD_Beneficialtype
beneficial	Beneficial	INT	X	RJ > AC_REFERRAL.referrer_ID DO > TBD EX > Should be always 1
transactionother_amount	Amount	INT	X	Value is taken from System Settings: Referral_Bonus value

3. Actions available in the table: N/A
4. Table options:
    a. Sort by
        i. Transactionother_id (default)
        ii. Transaction_type
        iii. beneficial_type
    b. Filter/Search by
        i. transaction_type
        ii. beneficial_type
        iii. Beneficial

Charities

1. Charities are the entities which the user will donate his coins to.
2. Charities will be added to the system by Admin only.
3. In the Charity page, there will be an option to exchange coins from each charity to real money.
4. In the Charity component we will manage 2 tables: CH_ENTITIES, CH_TRANSACTIONS.
5. Charity fields to be displayed in the table:

Charity > CH_ENTITIES

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
charity_id	ID	INT	X	AUTO_INCREMENT
charity_logo	Logo	File upload	Y	Upload
charity_name	Name	VARCHAR(100)	Y	Text
charity_incoins	In	INT	Y	Feed from Transactions tables
charity_outcoins	Out	INT	Y	Feed from Transactions tables
charity_balance	Balance	INT	Y	= charity_incoins - charity_outcoins
charity_real	Donated	INT	Y	= charity_balance exchanged to real money based on system settings: Exchange_rate
charity_withdraw	Withdraw Amount	TEXT	V	Manually defined
action	Donate	BUTTON	X	Withdraw charity_balance and charity_real by charity_withdraw amount. Trigger a new record to

				CH_TRANSACTIONS
charity_totaldonated	Total donated	INT	Y	SUM(ch_transaction.charity_withdraw)

1. Actions available in the table: Add,Edit (only editable fields),Delete, View
2. Table options:
    a. Sort by
        i. charity_id (default)
        ii. charity_name
    b. Filter/Search by
        i. charity_name

Charity > CH_TRANSACTIONS

Field Name	**Label**	**Type**	**Editable (system & manually)**	**Value source**
charitytransaction_id	ID	INT	X	AUTO_INCREMENT
charity_id	Charity ID	INT	X	ch_entities.charity_id
charity_name	Name	VARCHAR(100)	Y	ch_entities.charity_name
charitytransaction_month	Month	VARCHAR(20)	Y	The calendar month of the transaction
charity_withdraw	Amount	INT	Y	charity_entities.charity_withdraw

1. Actions available in the table: Add,Edit (only editable fields),Delete, View
2. Table options:
    a. Sort by
        i. charitytransaction_id (default)
        ii. charity_name
        iii. charitytransaction_month
    b. Filter/Search by
        i. charity_name
        ii. charitytransaction_month

System Settings

Pressing settings wheel icon on the top right corner of the screen will open a menu with the following items:

- System Tables and System Settings
- Dashboard & Statistics

System Tables and System Settings

TD_Itemtype

TD_Itemtype		
id	Value	Done
1	Task	1
2	Habit	21

Actions available in the table: Add, Edit, Delete

TD_Itemlevel

TD_Itemlevel					
id	Value	Itemtype	Coins	Referral	Limit
1	Silver	1	1	0	4
11	Silver	2	51	1	4
2	Gold	1	5	1	2
22	Gold	2	55	2	2
3	Diamond	1	10	2	1
33	Diamond	2	60	5	1

Actions available in the table: Add, Edit, Delete

TD_Eventtype

TD_Eventtype	
id	Value
1	Select
2	Done

Actions available in the table: Add, Edit, Delete

TD_Transactiontype

TD_Transactiontype	
id	Value
1	RJ
2	RC
3	IC
4	DO
5	EX

Actions available in the table: Add, Edit, Delete

TD_Beneficialtype

TD_Beneficialtype	
id	Value
1	Admin
2	User
3	Charity

Actions available in the table: Add, Edit, Delete

System Settings (app_param)

app_param	
exchange_rate	100
referral_limit	5
Referral_Bonus	20
Inactive_days	14

Actions available in the table: Add, Edit, Delete

Dashboard & Statistics

User app

Modules and Features

1. Settings Module
    - Text (can be translated?), symbols, icons, bars and toggle buttons
    - Settings options:
        - Language (Select a flag, it will impact the sentences language that will be populated)
        - Number of daily options to select from (1-4)
        - Sounds: On/Off toggle
        - Notifications: On/Off toggle
        - Invite a friend to use app (Will create a unique URL to support the pyramide reward model (See below) Link or QR code
2. **Registration Module**
    - Users register using their phone number or email address.
    - Required user details: Nickname, Avatar, Gender, Year of Birth.
3. **Random Sentences Module**
    - Each day, users receive random sentences (Following their preference) based on their registered gender and age.
    - Users select one sentence and confirm completion of the task described in the sentence.
    - Each sentence is associated with a difficulty level:
        - Easy - 1 point/coin
        - Medium - 5 points/coins
        - Hard - 10 points/coins
    - If a sentence was not marked as “Completed” it’s terminated by midnight.
4. **Statistics and Scoring Module**
    - Records sentences confirmed by the user including the date of completion.
    - Provides weekly and cumulative scoring for the user.
5. **Notification Module**
    - Sends reminders and motivational notifications to users for task completion.
6. **Settings Module**
    - Allows users to configure system settings such as language preference and notification schedules.
    - Follow section #1 for the required Setings
7. **Secure Management System**
    - Ensures secure storage of registered user details.
    - Tracks registration date, completed sentences count, and user scores.
8. **Language Support**
    - Supports multiple languages for user interface and notifications.

Technical Requirements

- The application should be developed using cross-platform frameworks such as React Native or Flutter to ensure compatibility across iOS and Android devices.

- Data storage should be managed securely, adhering to industry standards for encryption and user data protection.

Conclusion

This specification outlines the core functionalities and technical requirements for the development of a mobile application aimed at enhancing user engagement through daily interactive tasks. By implementing these features, the application aims to provide a personalised and rewarding user experience across different mobile platforms.

Pyramid Reward Model with Difficulty Levels

Level 1: Direct Invites

- When a user invites a friend directly, they earn **10 points**.

Level 2: Invites by Invited Friends

- If the friend (invited in Level 1) further invites someone else, the original user earns **5 points**.

Level 3: Invites by Friends of Invited Friends

- If the friend invited in Level 2 invites another person, the original user earns **2 points**.

Difficulty Levels for Actions:

Actions performed by users can vary in difficulty, influencing the points earned:

- **Easy Action:** Users performing an easy action earn **1 point**.
- **Medium Action:** Users performing a medium difficulty action earn **3 points**.
- **Hard Action:** Users performing a hard difficulty action earn **5 points**.

Example Scenario Incorporating Difficulty Levels:

1. **User A** invites **Friend B** directly and performs an easy action.
    - User A earns 10 points (invite) + 1 point (easy action) = **11 points**.
2. **Friend B** invites **Friend C** and performs a medium action.
    - User A earns 5 points (invite by Friend B) + 3 points (medium action) = **8 points**.
3. **Friend C** invites **Friend D** and performs a hard action.
    - User A earns 2 points (invite by Friend C) + 5 points (hard action) = **7 points**.

How It Works:

- Each invite contributes to the pyramid structure, where points diminish with each level of separation from the original user.

- Actions are classified into different difficulty levels (easy, medium, hard), with corresponding point values assigned to each level.
- Users earn points based on both their invite actions and the difficulty level of the actions they perform.
- This complexity encourages users to not only invite others but also engage in more challenging actions to earn higher rewards, thereby increasing engagement and participation.

Considerations:

- Adjust the points awarded for each level and difficulty level according to your specific goals and user engagement strategies.
- Monitor user behavior to ensure that the difficulty levels and points align with the desired user actions and motivations.
- Communicate clearly with users about the point structure to enhance transparency and encourage participation.

This model provides a structured approach to rewarding users based on both their invitation efforts and the difficulty of actions they perform, adding depth and complexity to your reward system.