FINAL ASSIGNMENT DOCUMENTATION
WEB PROGRAMMING

ConnectFriend – Job Friends

Made by:

Name : Ivan Adrian

NIM : 2602076323

Class : LC01

Link Github: https://github.com/ivanadriannn/FinalASGWebProg-2602076323-Ivan-Adrian

COMPUTER SCIENCE

BINA NUSANTARA UNIVERSITY JAKARTA
2025

I.   Backend Logic

Models

1. **Avatar Model**:
   The Avatar model represents the customizable avatar images that users can choose, purchase, and set as their profile pictures in the application.
2. **FieldOfWork Model**:
   The FieldOfWork model stores the various job sectors or professions that users can associate with their profile, helping them find connections in the same field.
3. **Friend Model**:
   The Friend model tracks the relationships between users by storing the pairing of user IDs to their respective friends, enabling them to become mutual connections within the app.
4. **Message Model**:
   The Message model handles the storage of text messages exchanged between users, supporting the communication feature in the social network.
5. **Notification Model**:
   The Notification model keeps track of notifications sent to users, such as friend requests or messages, and links them to both the receiving user and the sender.
6. **Profession Model**:
   The Profession model represents a user's specified profession, typically for those searching for job-related connections, helping with filtering and linking users by their professional background.
7. **Transaction Model**:
   The Transaction model records payment-related transactions, such as registration or purchases made by the users, managing and tracking the financial aspect of the platform.
8. **User Model**:
   The User model represents the authenticated users of the application, containing their personal data, relationships, fields, and ensuring secure authentication via Laravel's built-in functionality.
9. **UserField Model**:
   The UserField model connects users to their selected fields of work, allowing for multiple areas of expertise or job interests for a given user to be stored and associated.

Relationship

1. professions table and users table
   - Relationship: *One-to-Many*
   - Explanation: The users table has a profession_id foreign key referring to professions.id. This indicates that each user has one profession, but a single profession can be held by many users.

2. users table and user_fields table
   - Relationship: *One-to-Many* (from Users to UserFields)
   - Explanation: The user_fields table has a user_id foreign key referring to users.id and a field_of_work_id foreign key referring to field_of_works.id.

This implies that a user can have many work fields through the user_fields table, and each work field (field_of_works) can belong to many users.

3. users table and friends table
- Relationship: *Many-to-Many*
- Explanation: The friends table acts as a junction table with two foreign keys to the users table: user_id and friend_id. This establishes that users can be friends with one another, and this relationship can exist between multiple user pairs (many-to-many). Additionally, sender_id indicates who sent the friend request.

4. users table and messages table
- Relationship: *One-to-Many*
- Explanation: The messages table contains two foreign key columns, sender_id and receiver_id, which refer to users.id. This means that each message is sent by one user (sender) to one other user (receiver), thus forming a one-to-many relationship (many-to-one from the perspective of both the sender and receiver).

5. users table and notifications table
- Relationship: *One-to-Many*
- Explanation: The notifications table has both a user_id and sender_id foreign key columns, both pointing to users.id. This implies that each notification is associated with one recipient user (user_id) and one sender user (sender_id), meaning that each user can receive multiple notifications.

6. users table and transactions table
- Relationship: *One-to-Many*
- Explanation: The transactions table contains a user_id foreign key referring to users.id and an avatar_id foreign key referring to avatars.id. This means that each user can have many transactions related to the avatars they purchase, resulting in a one-to-many relationship from users to transactions.

7. avatars table and transactions table
- Relationship: *One-to-Many*
- Explanation: The transactions table also contains an avatar_id foreign key referring to avatars.id. This suggests that each avatar can be purchased multiple times by different users, creating a one-to-many relationship from avatars to transactions.

8. avatars table and users table
- Relationship: Indirect connection through transactions table
- Explanation: There is no direct relationship between avatars and users without involving the transactions table. The relationship between a user and an avatar is mediated through a purchase recorded in the transactions table.

Initial Home Page:
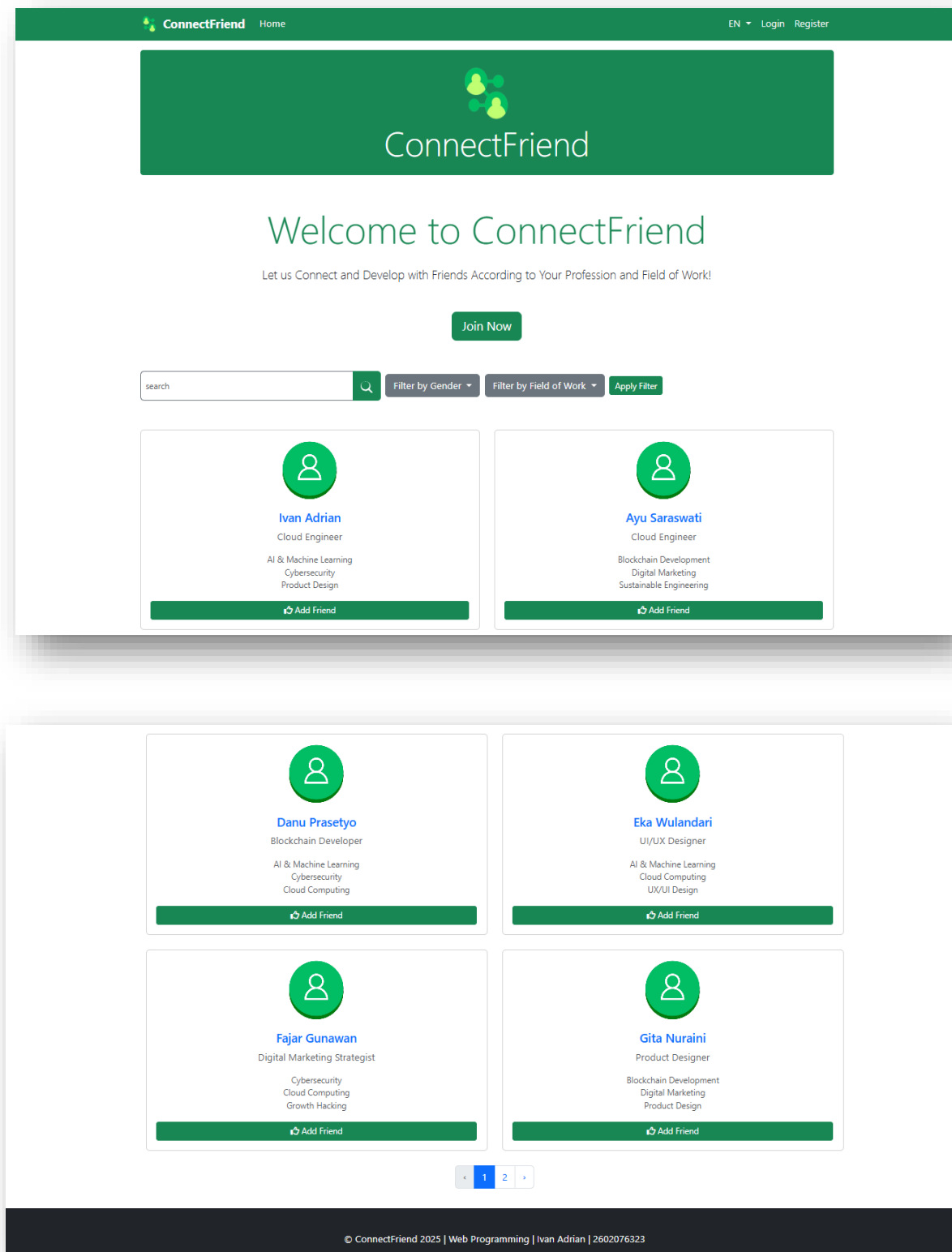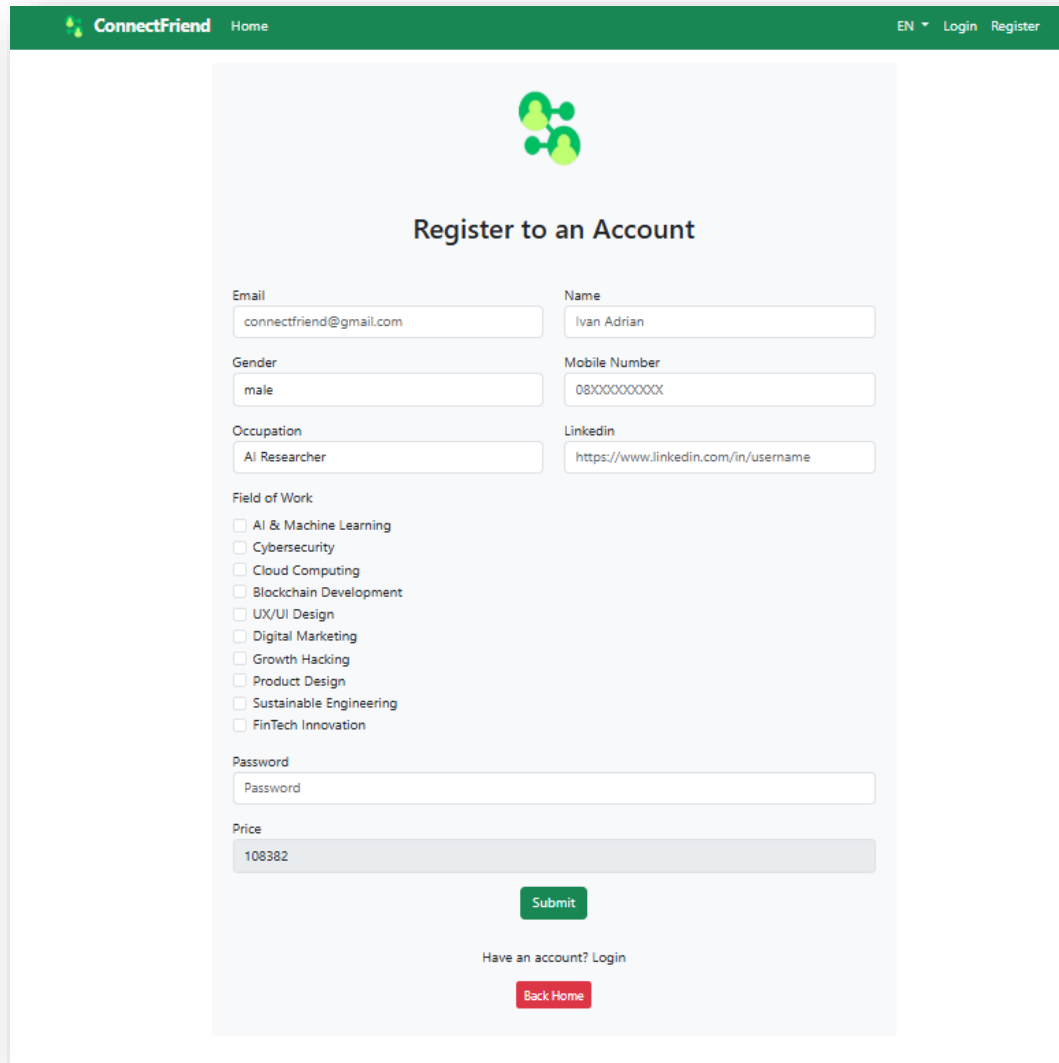
This page is the first display which is the welcome page when the user uses the website. Here the user's access is still limited because as a guest role he uses middleware with a combination of auth middleware and sessions. As guests, they can only log in, register, and also view the list of

available users and search and filter users with the same profession or field of work. If the user clicks one of the lists of users, they will go directly to the login page.

Main Features:
1. **[LO1,LO2,LO3 & LO4, 20 points] User Authentication**

Register

On this page, users can register an account by filling in several required things such as
a) Email
b) Name
c) Gender
d) Mobile Number
e) Field of Work
f) LinkedIn
g) Password

Each existing field also has its own validation attached in the image below.

Register Validation

Login

When the user has finished the registration, they can log in using their email and password. Login also has validation when the fields have not been filled in, the data does not match, or when the login fails.

Login Validation

2. **[LO1,LO2,LO3 & LO4, 10 points] Payment**
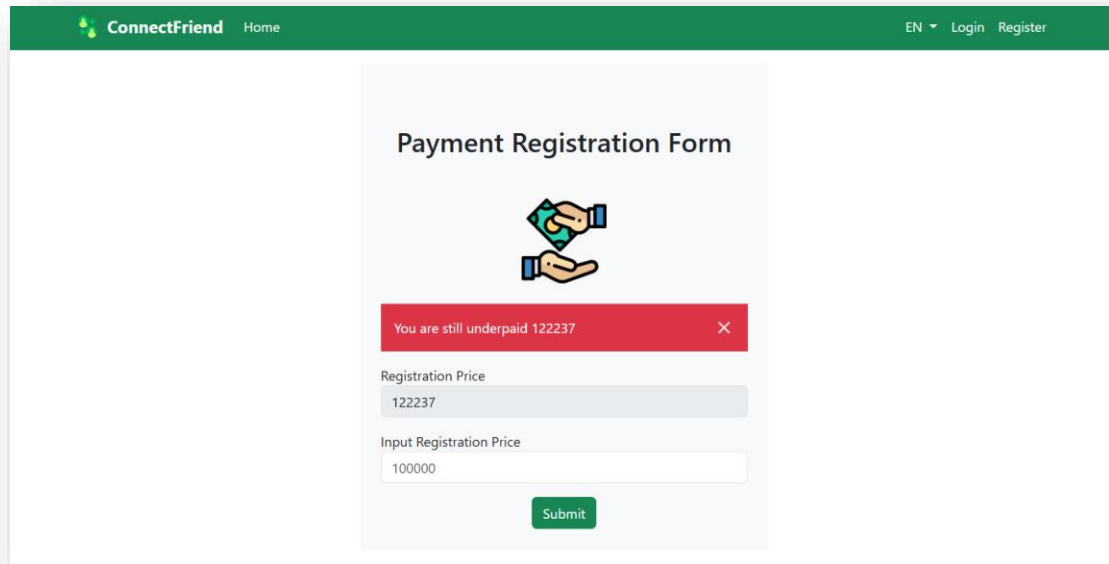
Payment

When all user data has been filled in during registration, they will be directed to the random registration fee payment page directly from the system.
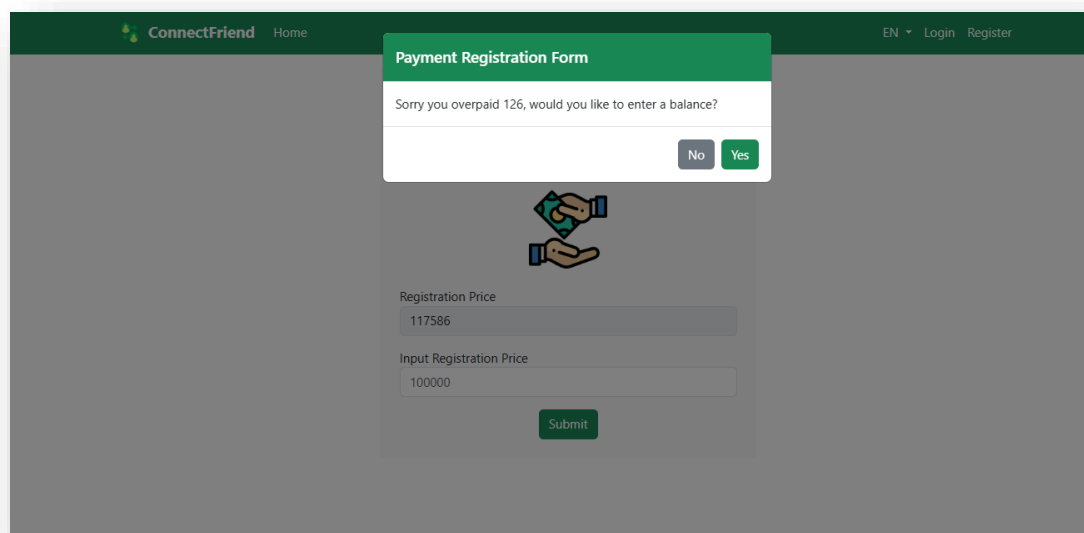
Underpaid Payment

Popup modal will appear when there is an overpayment, the user has two choices, namely topping up the registration price or adding the excess coins to the user's own coin balance.

Overpaid Payment

3. **[LO1,LO2,LO3 & LO4, 30 points] Friends**

The following is the first page when finished logging in, the user can see friends who have certain fields of work, and add them to the friends list by clicking the add friend thumb button. If so, the button status will be "Pending" and add the friends into the pending wishlist. If it has been accepted, the button will change to "Remove".
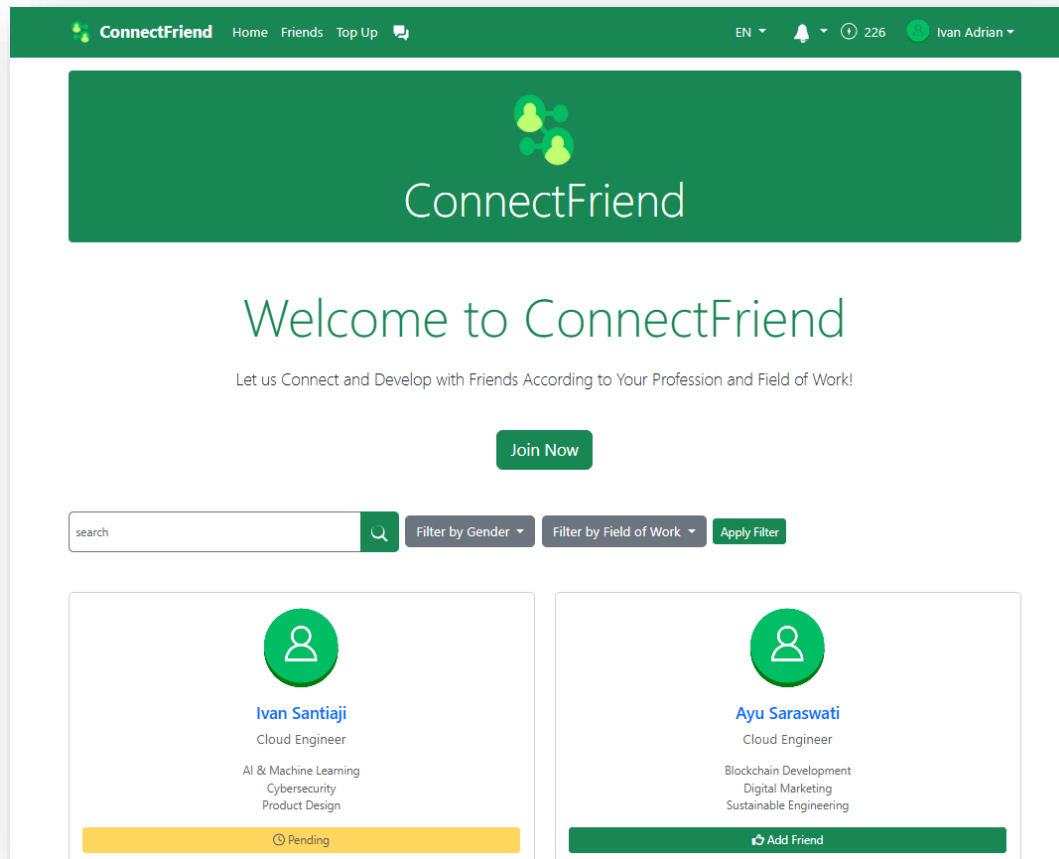
Wishlist Pending Friend Request
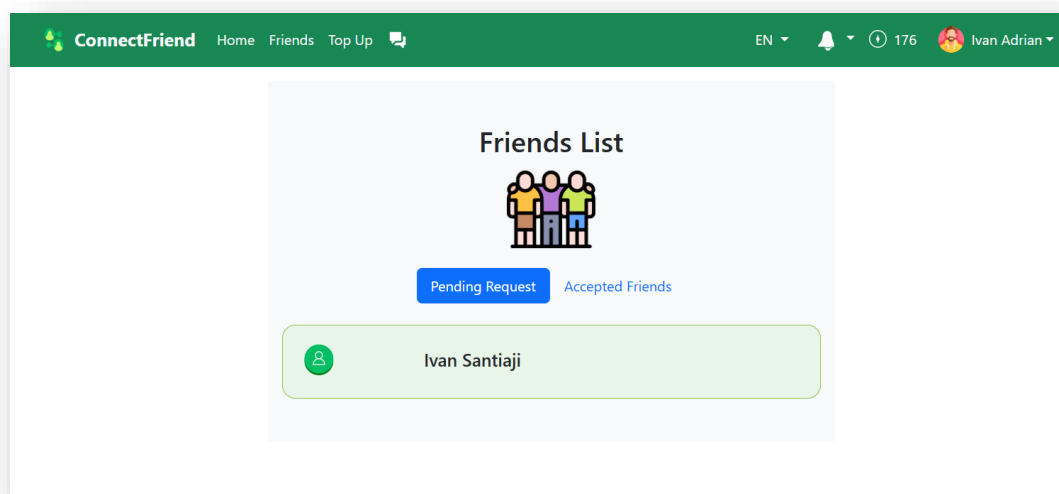
The following is a page for accepting or rejecting friend requests from users who have the same field of work.

Friend Request
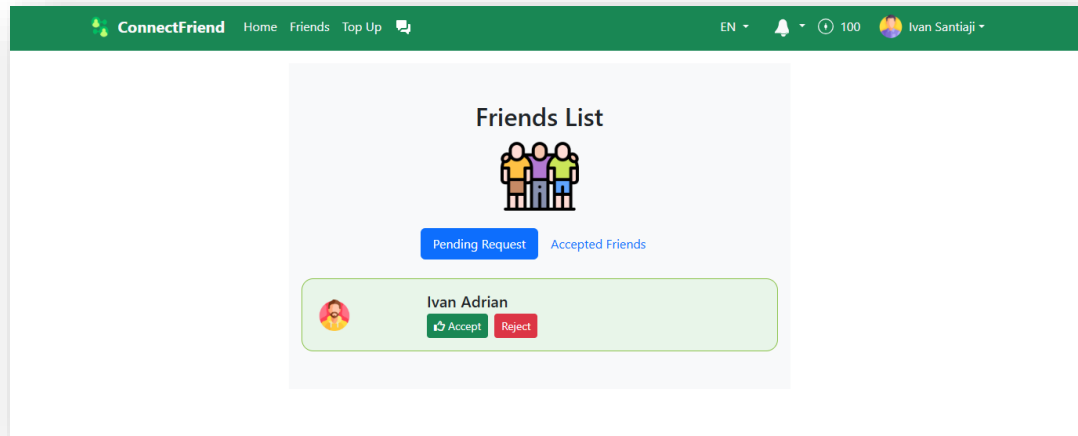
If you have successfully accepted new friends, you will move to the list of accepting friends. Here, users can start conversations or remove friends.

Accepted Friends
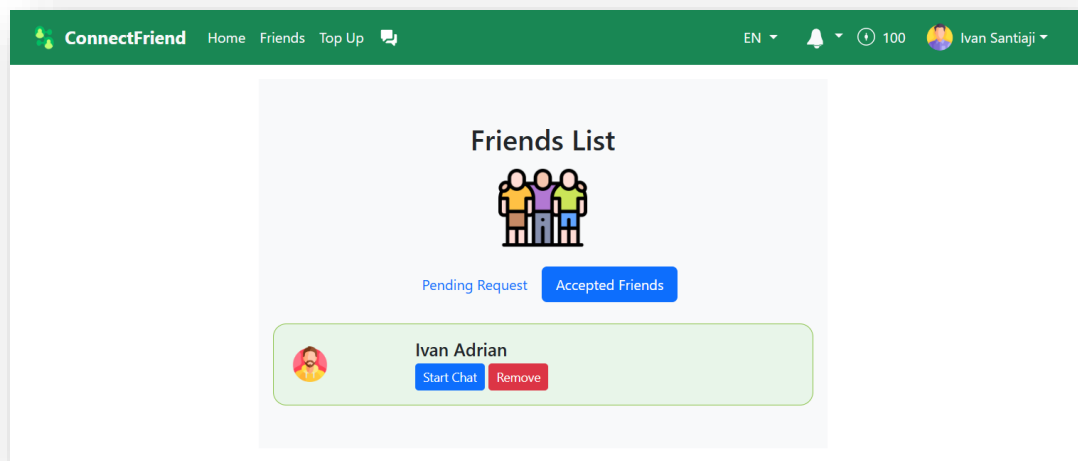
4. **[LO1,LO2,LO3 & LO4, 20 points] Messaging/Video Call**

Chat
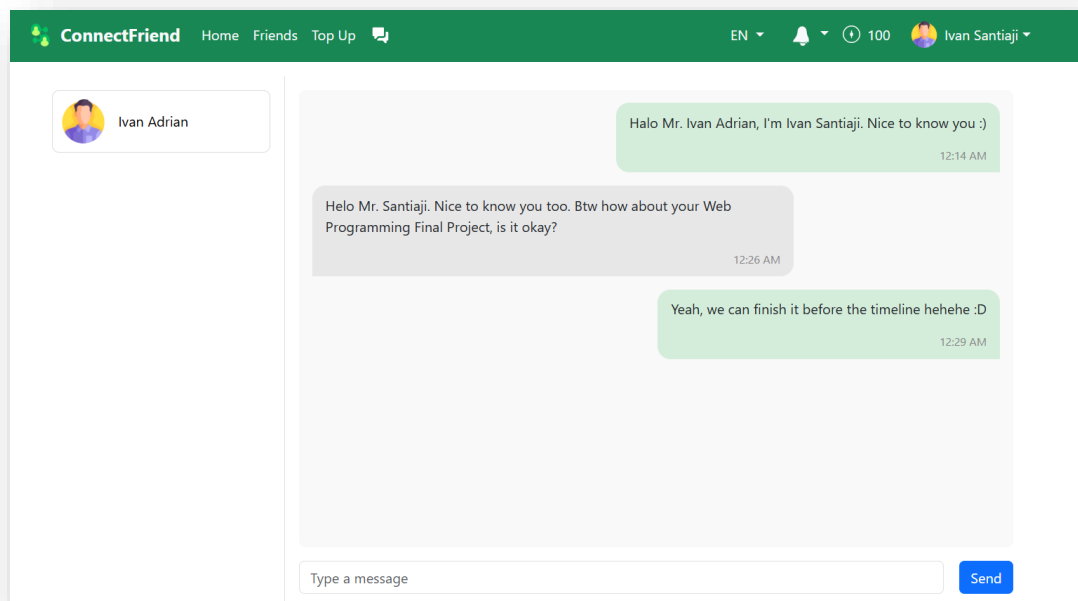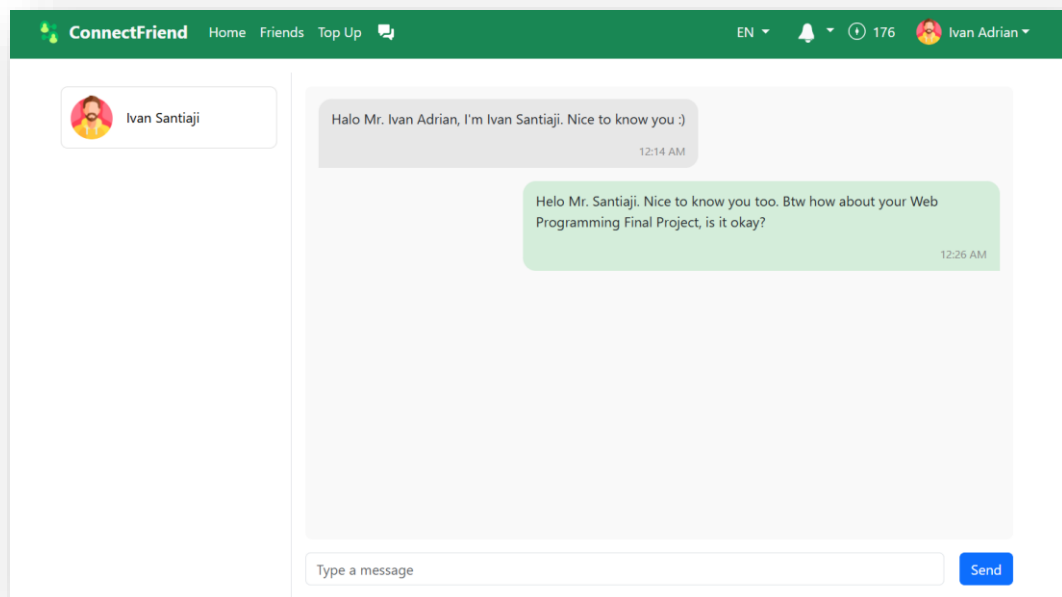
5. **[LO1,LO2,LO3 & LO4, 10 points] Searching & Filtering**

Searching (Occupation) & Filter (Gender and Field of Work)

Searching (Field of Work) & Filter (Gender and Field of Work)

Searching (Name) & Filter (Gender and Field of Work)

Filter (Gender and Field of Work)

Searching (Name/Occupation/Field of Work)

6. **[LO1,LO2,LO3 & LO4,10 points] Notification**
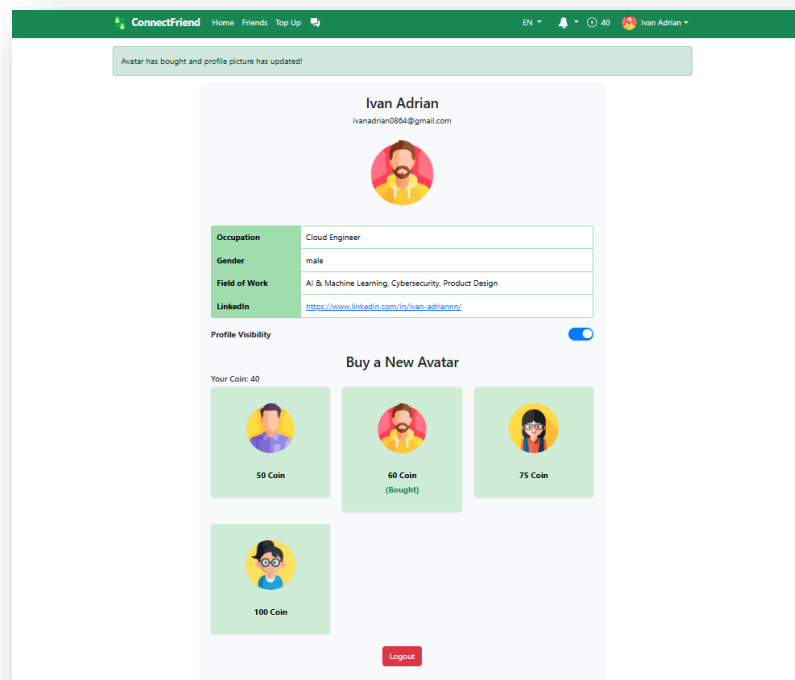
Notification from chat or friend request
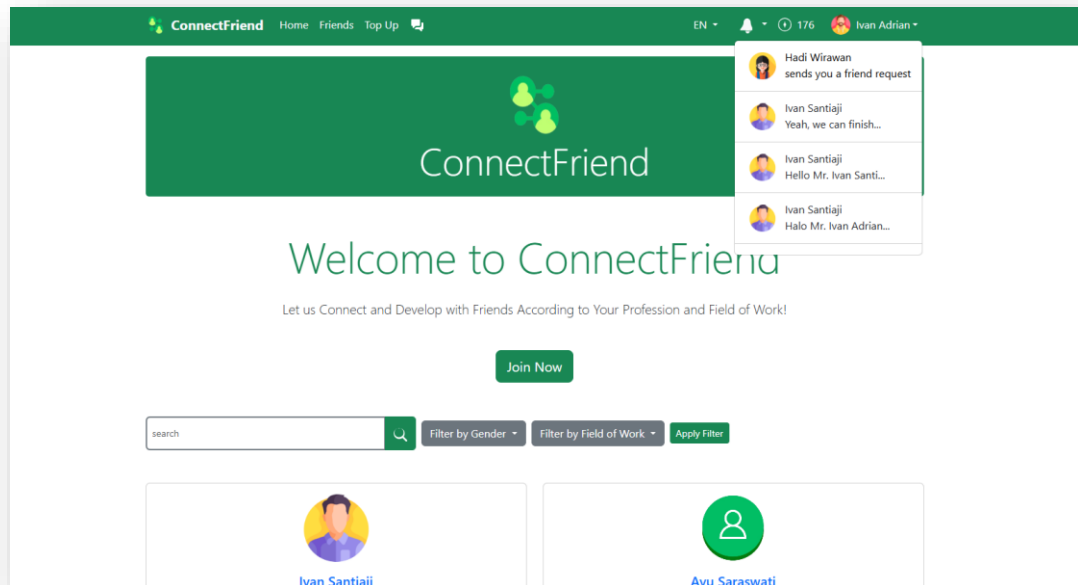
Additional Features:

7. **[LO1,LO2,LO3 & LO4, 30 points] Buy Avatar**
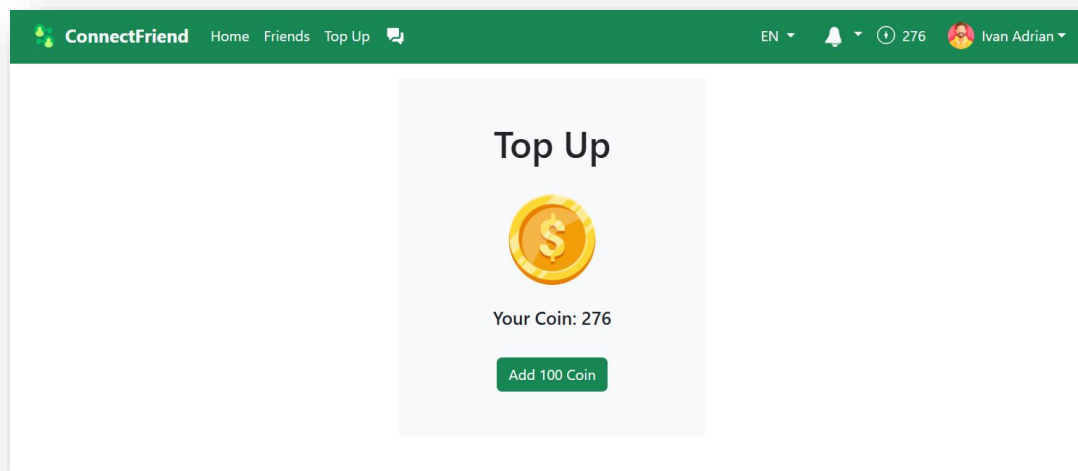
Avatar Market
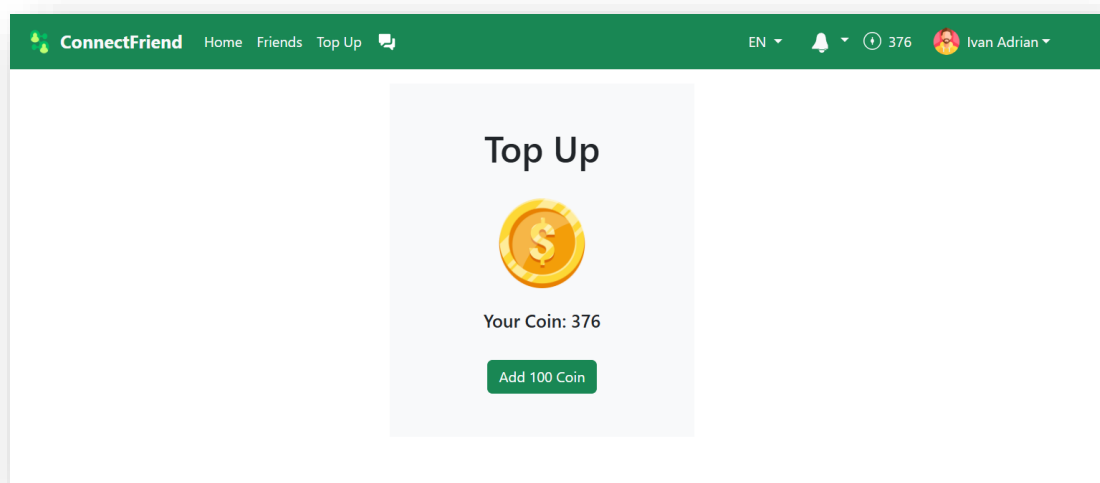
Inssuficient Coins to Buy Avatar

8. **[LO1,LO2,LO3 & LO4, 15 points] Topup Coins**
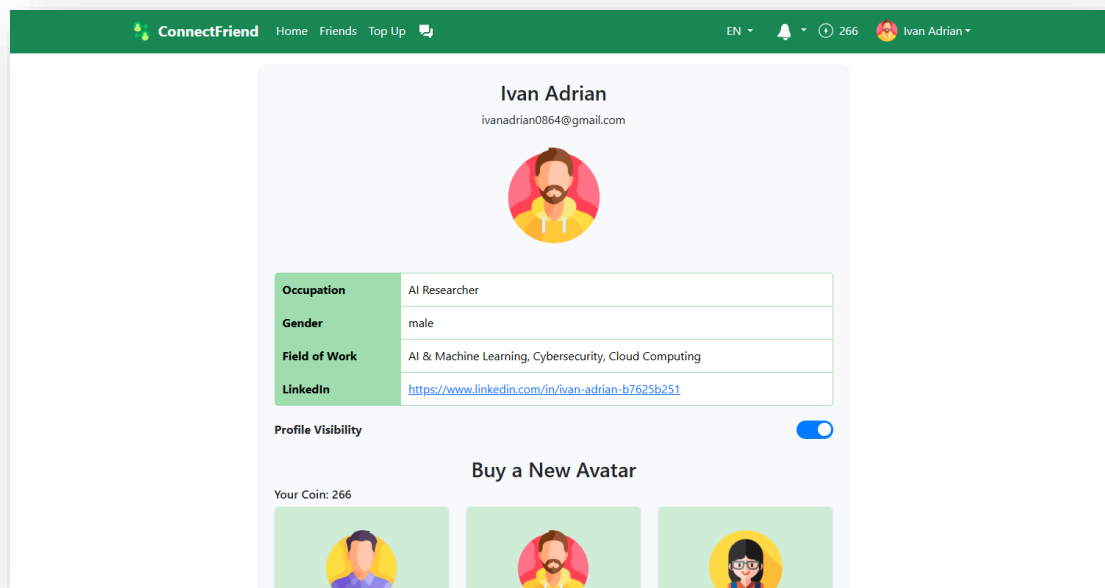
Topup coins

Display if coins added successfully. Every time you click the add coin button, 100 coins will automatically be added.
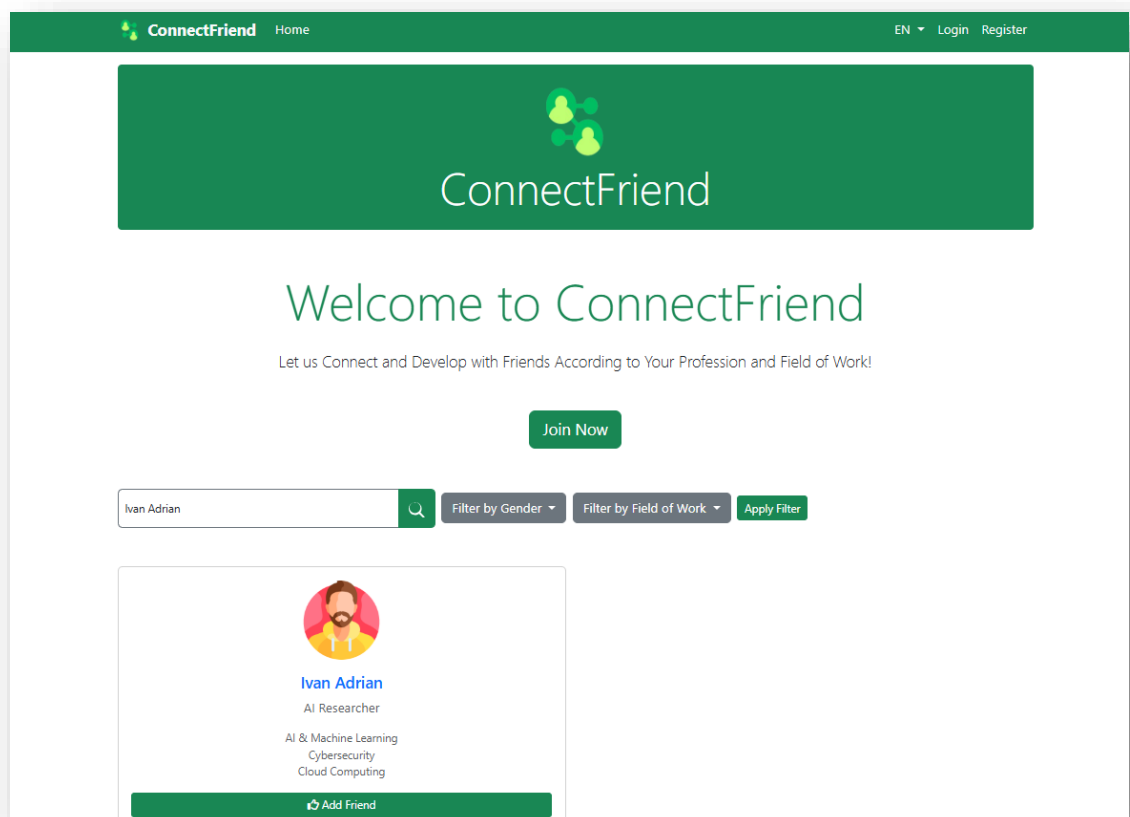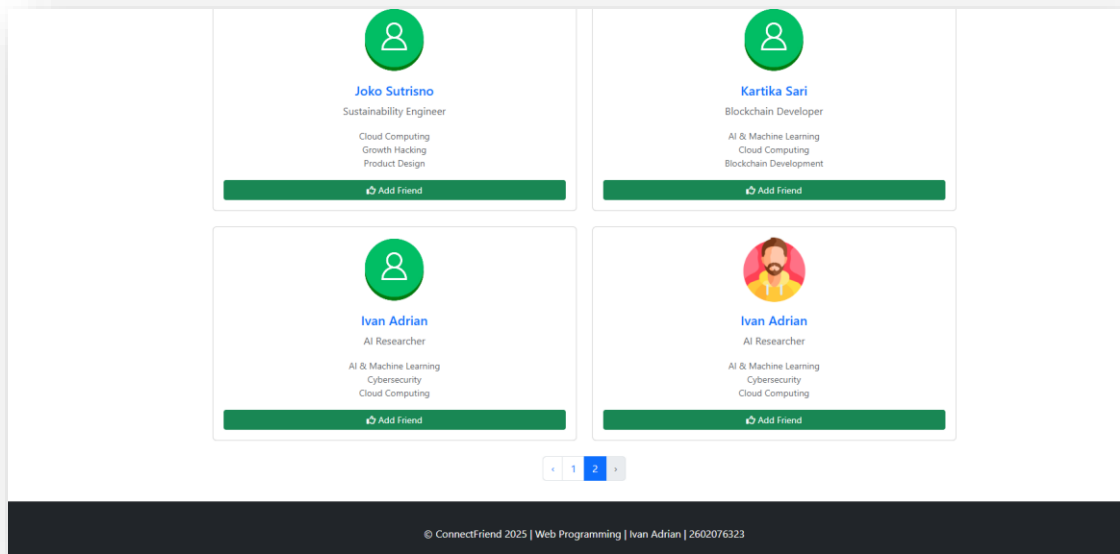
9. **[LO1,LO2,LO3 & LO4, 30 points] Visible Profile**
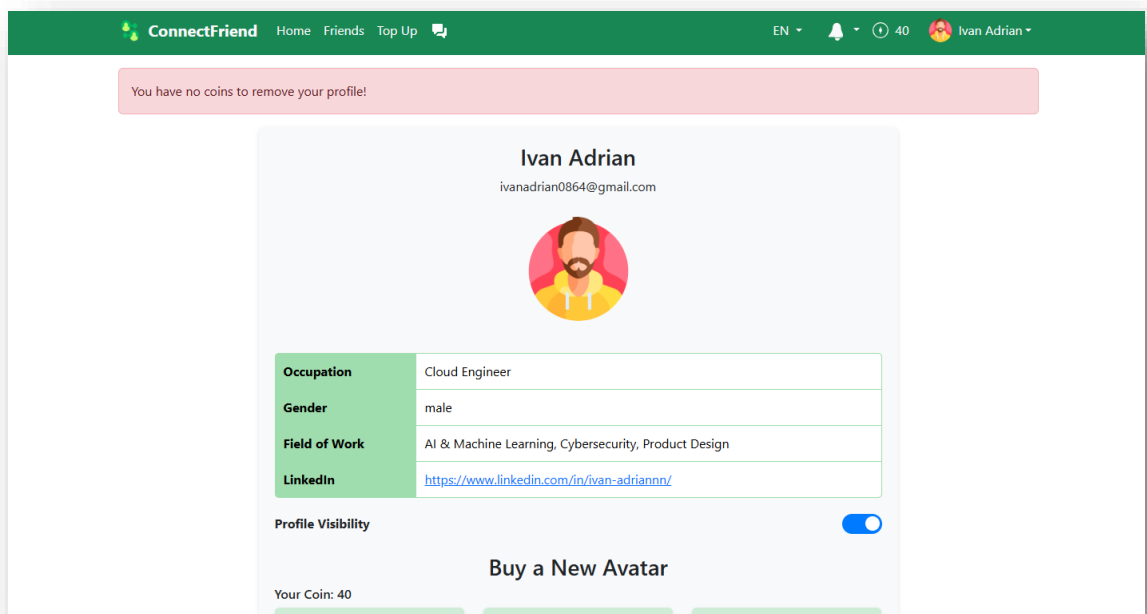
Visible Profile

Before going invisible, we check the homepage and search for the username "Ivan Adrian", then we can see that the user profile is still visible.

Then to change the profile to invisible, just press the toggle switch, if the balance is sufficient 50 coins, then the profile will become invisible, and vice versa if the balance is less then an error message will appear at the top of the page.

If the user profile "Ivan Adrian" has been successfully made invisible, a message will appear at the top and the profile image is changed to be a random bear. The profile will automatically not be visible to other users who search.
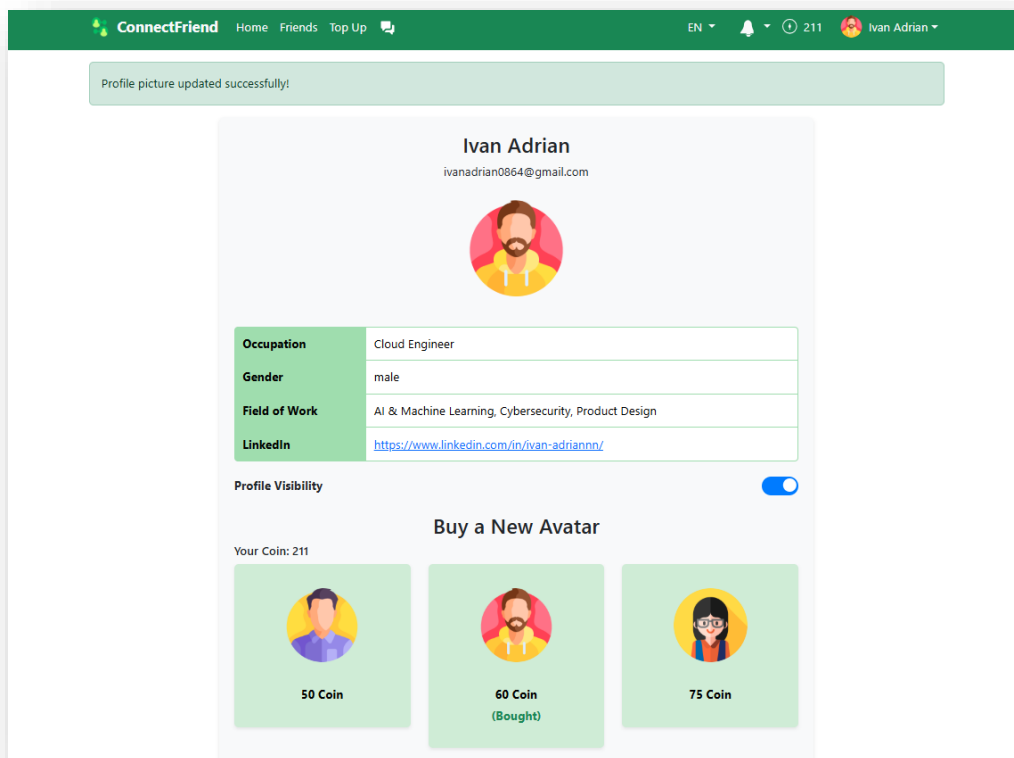
Invisible Profile
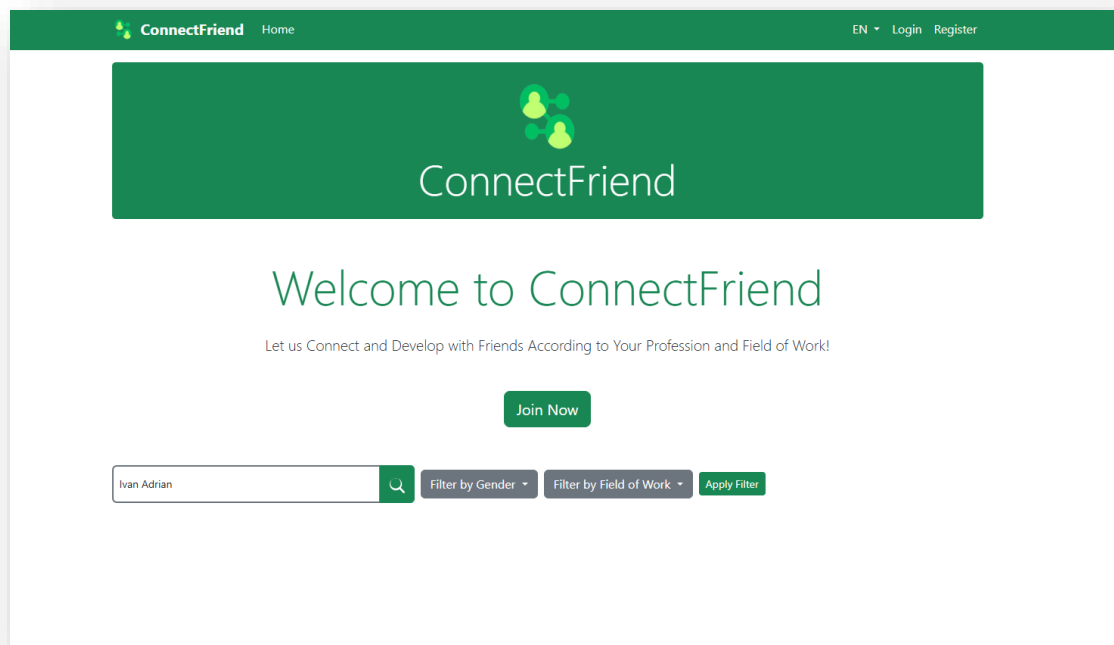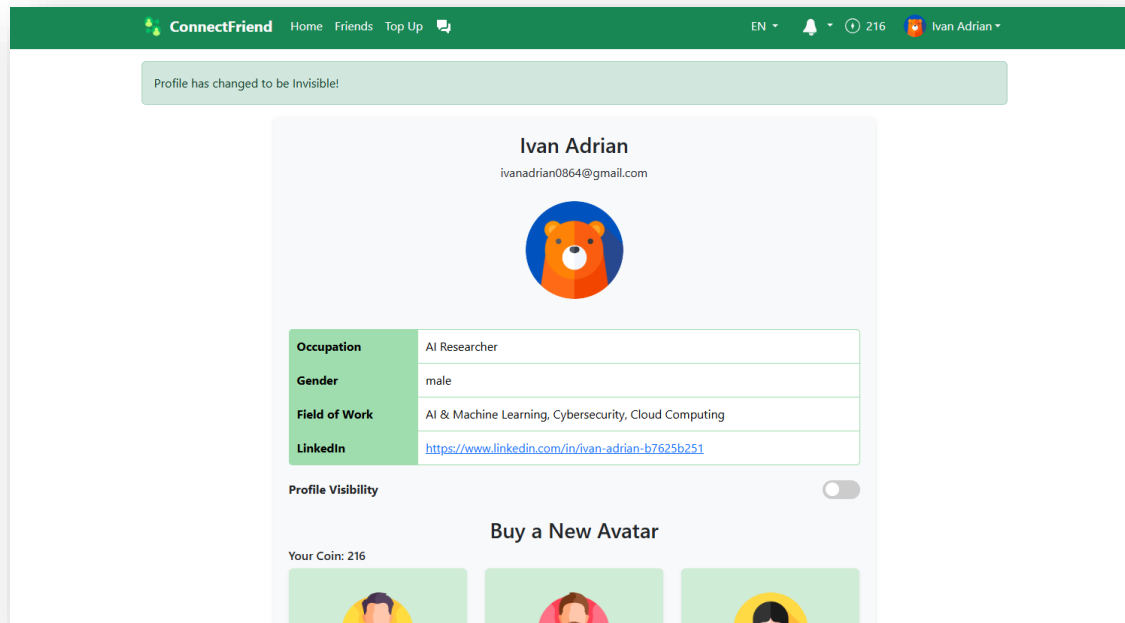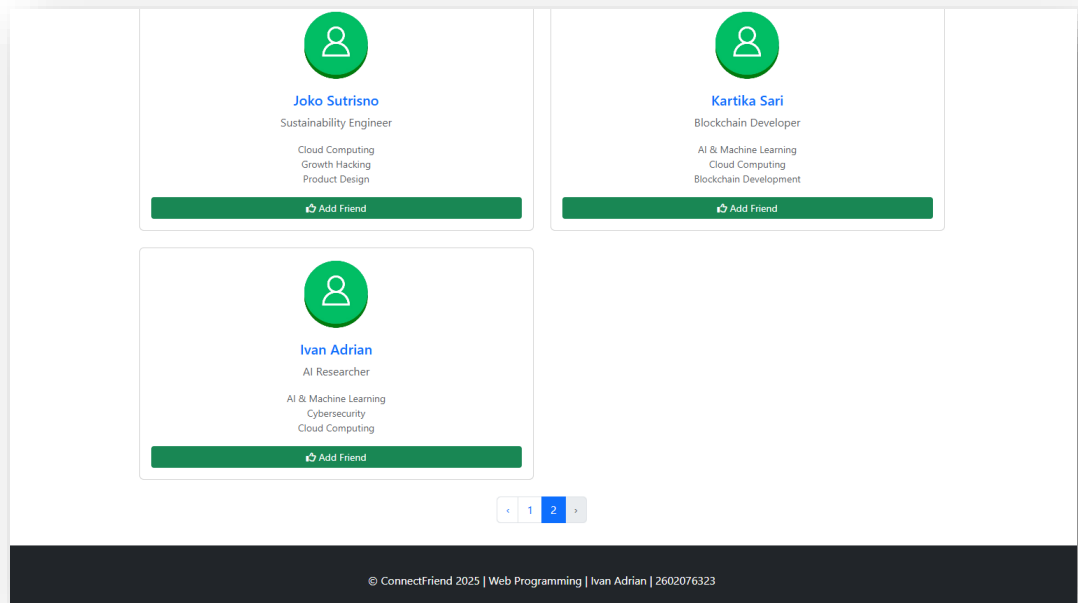
If you want to restore your profile, just log back into your account, then go to the profile page, and click the toggle switch again, this change will cost 5 coins, if there are enough coins then the user's profile image will return to the default and the account will be available again in the profile other users if searched. When it becomes visible again, there will be a success message at the top.
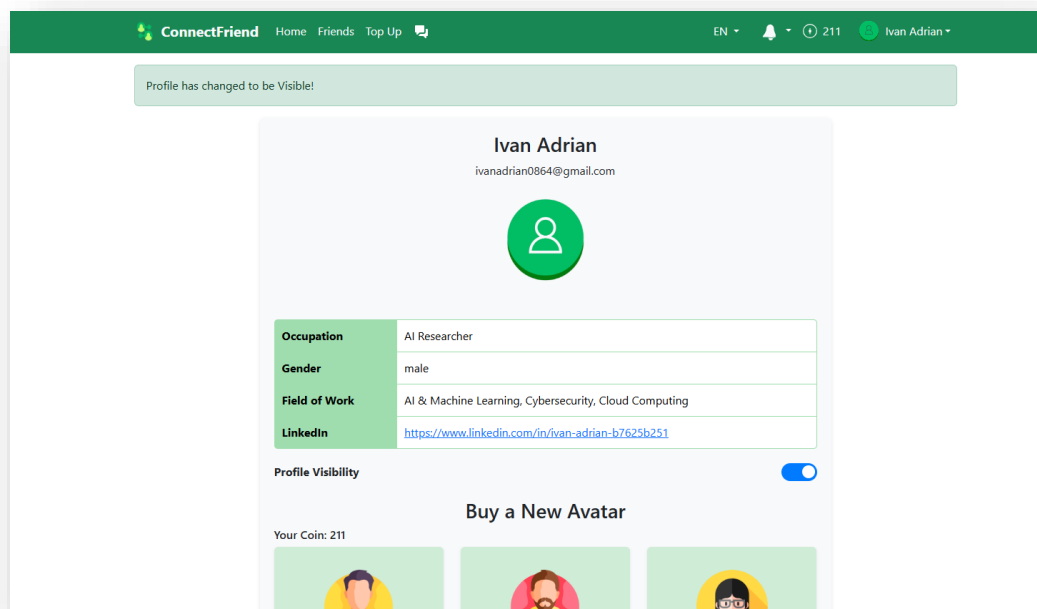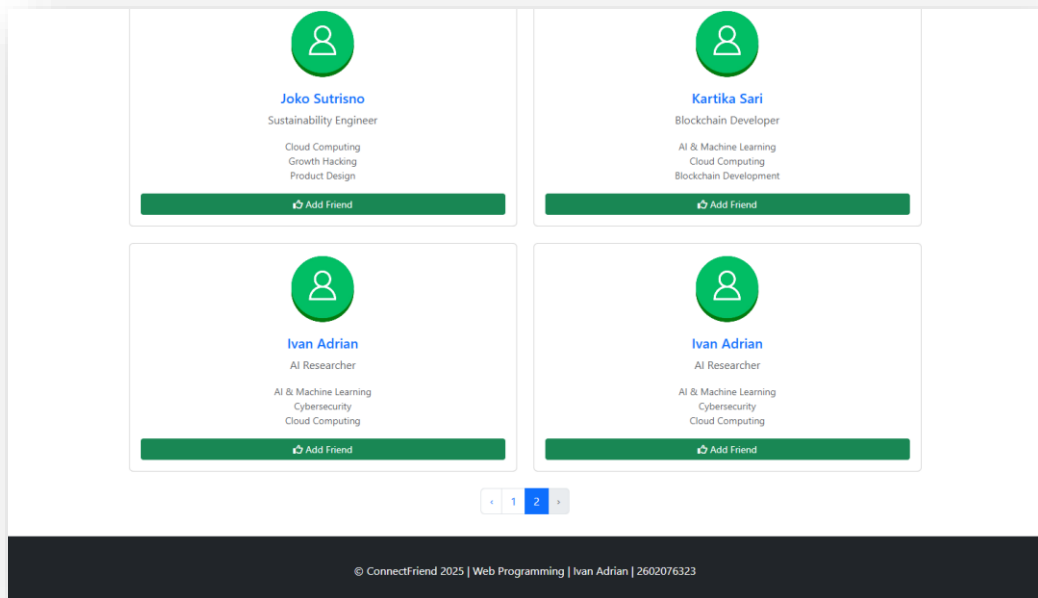
Then we can update the profile image again with the previously purchased avatar.

Localization