Second Term Project - Messaging Application / Symfony

MohammedAmineAghlaOuaziz

P1. Messaging Web application

Introduction

The project consists of creating a web application that allows users to exchange messages. It is similar to the usual internal messaging systems in companies or a webmail system.

Requirements Specification for Symfony Messaging Application

The Symfony Messaging Application is intended to serve as a robust and secure platform for users to communicate effectively. Accessible via a username-password combination securely stored in a database, the application facilitates seamless messaging between users. The following outlines the key features and functionalities of the application:

Authentication and User Management

1. **User Registration and Login**:
    - Users can register with unique usernames and passwords.
    - Passwords will be securely hashed and stored in the database.
    - Upon registration, users will receive a confirmation email to activate their accounts.
2. **Password Recovery**:
    - Users can request password recovery via email.
    - A secure password recovery process will be implemented, ensuring the confidentiality of user data.

Messaging Features

1. **Message Sending**:
    - Users can compose and send messages to specific recipients.
    - Support for multiple recipients in a single message.
2. **Inbox Management**:
    - Users can view a concise overview of received messages in their inbox.
    - Full message content can be viewed individually when entering the desired message
3. **Outbox Management**:
    - Sent messages will be organized in the outbox, detailing the recipient's interaction status.
    - Users can track whether messages have been read or remain unread by the recipients.

Advanced Features

1. **Attachment Support**:
    - Users can attach images and other files to their messages.
    - Images sent by the sender are viewed directly without the need to download them.
2. **Security Enhancements**:
    - Passwords will be encrypted using strong encryption algorithms.
    - User sessions will be managed securely to prevent unauthorized access.

Administrative Capabilities

1. **Admin Dashboard**:
    - Administrator will have access to a dashboard for user management and monitoring (block/unblock other users).
2. **Database and Server Connectivity**:
    - Connection data to the database and mail server will be securely stored and managed.
    - Configuration files will ensure the secure handling of sensitive information.

Main functionalities

A1: Access: Users access the application with a username and password that are stored in a database.

A2: Sending: Application users can send messages to other users (the recipient is chosen by name).

A3: Inbox: Users will be able to see a list or table with the messages that have been sent to them, differentiating between read and not. In the inbox the complete messages are
not read, there is only the subject or the beginning of the message.

A4: Messages: They will also be able to consult each message individually. Here if the full content is read and there is also the opportunity to respond.

A5: Outbox: The user will be able to see a table with his sent/her messages. Messages will be sorted by date sent, most recent first. In the table you must see the recipient ofthe message. From this table, users must be able to access each message. Users willbe able to see if the recipient has opened the message or not.

A6: The connection data to the database and to the mail server must be stored in files.

Extensions

E1. Encrypted user password

- Store the user passwords using an encryption method.

E2. Self-registration

- Realistic process for users to register on the website.
- To get the full score, you have to include an email with an activation link.

E3. Password recovery

- Realistic process for users to retrieve their password or generate a new one.
- If the solution is to send a new key (or the current one), without having to perform more operations, only half the score will be obtained.

E4. Profile image.

- Users will be able to upload (and update) an image to use as a profile image.
- It has to be visible to other users.

E5. Messages to multiple recipients

- Allow to enter more than one recipient in the message submission form.

E6. Administration zone

- Develop a zone only visible to some users (depending on their role).
- In this area you have to implement tasks typical of an administrator. For example, block or unblock users.

E7. Attached files

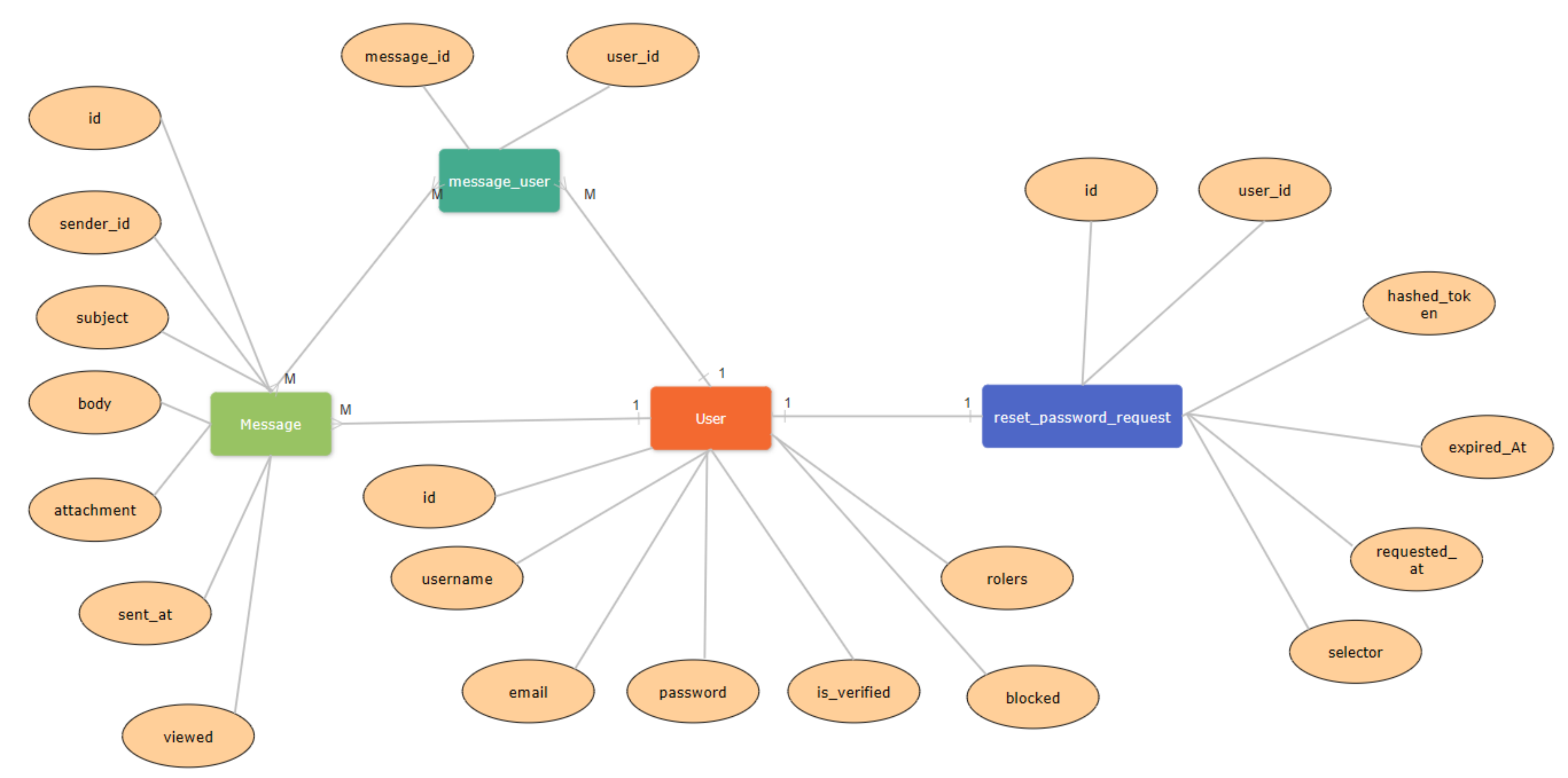
- Allow to attach files to a message.

E8. Images

- Allow to insert images as part of a message.
- For this extension to be considered correct, the images must be viewed directly when viewing the message, not as attachments or following a link.
- If the application does not allow sending to more than one user, in this extension only half of the score can be obtained.

We used Bootstrap library, to style the pages of the project for a good user experience.

E/R SCHEME



💡 ***Relationships*** :

-

Many-to-One: Each message is sent by one user (message.sender_id → user.id).

- One-to-Many: Each user can send multiple messages (user.id → message.sender_id).

- Many-to-Many: Messages can have multiple recipients (via the message_user table).

- One-to-One: Each user can have at most one active password reset request (user.id → reset_password_request.user_id).

Logical Model

The logical model for this messaging application comprises three primary entities: `User`, `Messages`, and `Password_Reset_request .` Each entity is defined by specific attributes, capturing essential information about users, messages, and password reset entries. The relationships between these entities establish the connections necessary for the application to function cohesively.

1. User:

- **Attributes:**
  - `id: INT` (Primary Key, Auto-Increment, Not Null)
  - `username: VARCHAR(50)` (Unique, Not Null)
  - `password: VARCHAR(255)` (Not Null)
  - `email: VARCHAR(100)` (Unique, Not Null)
  - `profile_Picture_filename: VARCHAR(255)` (Nullable)
  - `activation_token: VARCHAR(64)` (Nullable)
  - `blocked: ENUM('inactive', 'active')` (Default 'active', Not Null)
  - `roles:` JSON array field storing user roles (e.g., **["ROLE_USER"]**, **["ROLE_ADMIN"]**).

2. Message:

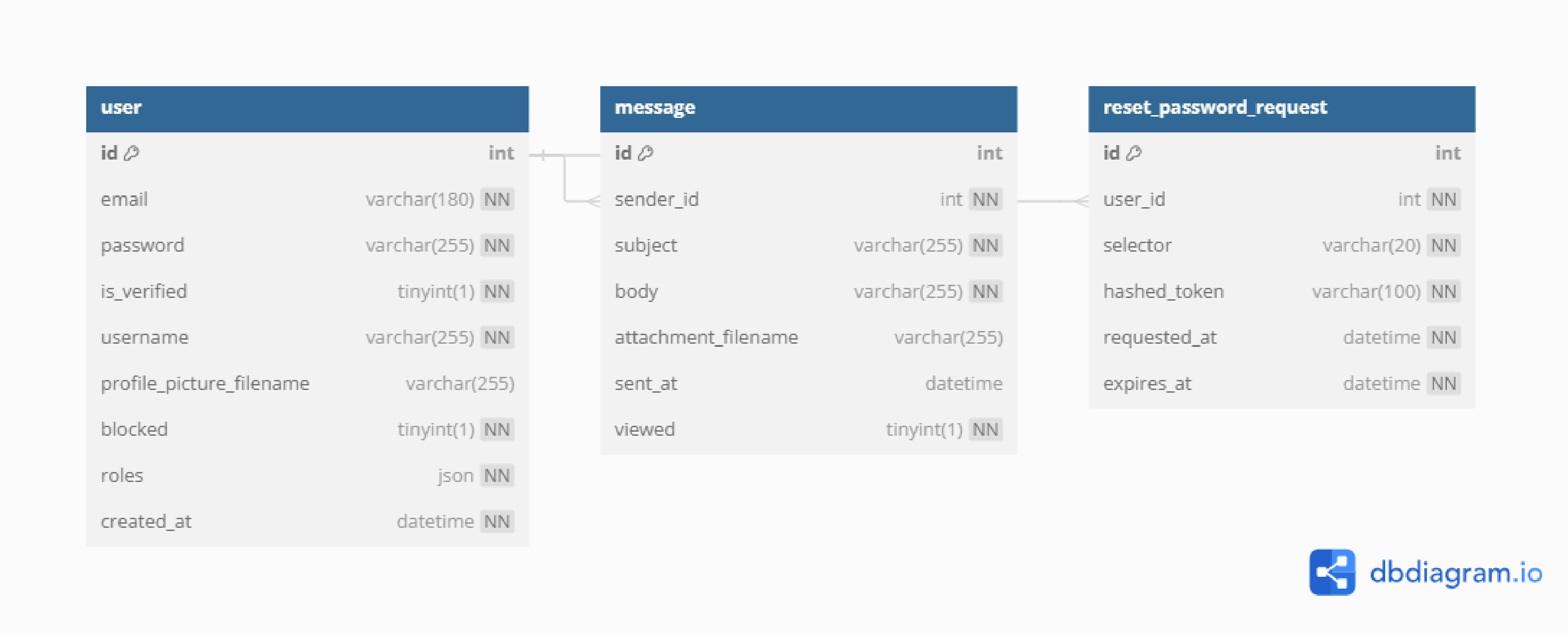
- **Attributes:**
  - `id: INT` (Primary Key, Auto-Increment, Not Null)
  - `sender_id: INT` (Foreign Key referencing Users, Not Null)
  - `subject: VARCHAR(255)` (Not Null)
  - `body: TEXT` (Not Null)
  - `attachment_filename: VARCHAR(255)` (Nullable)
  - `file_path: VARCHAR(255)` (Nullable)
  - `viewed: BOOLEAN` (Not Null)
  - `sent_at: TIMESTAMP` (Not Null)

3. reset_password_request:

- **Attributes:**
  - `id: INT` (Primary Key, Auto-Increment, Not Null)
  - `user_id: INT` (Foreign Key referencing Users, Not Null)

- `requested_at :` datetime (Not Null)
- `expires_at :` datetime (Not Null)
- `hashed_token: VARCHAR(255)` (Not Null)
- `expiration: TIMESTAMP` (Not Null)

Database Diagram



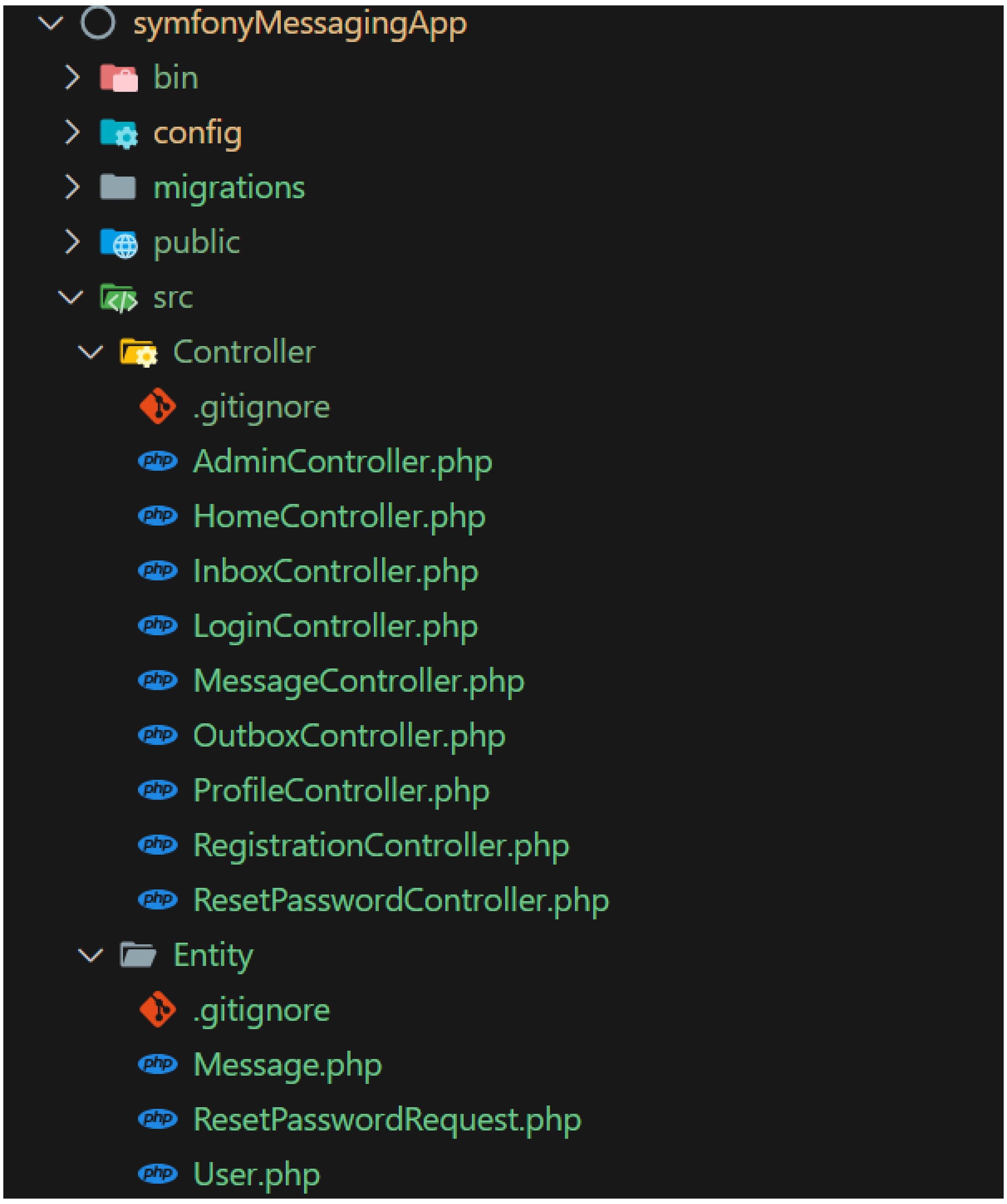
user		
id	int	
email	varchar(180)	NN
password	varchar(255)	NN
is_verified	tinyint(1)	NN
username	varchar(255)	NN
profile_picture_filename	varchar(255)	
blocked	tinyint(1)	NN
roles	json	NN
created_at	datetime	NN

message		
id	int	
sender_id	int	NN
subject	varchar(255)	NN
body	varchar(255)	NN
attachment_filename	varchar(255)	
sent_at	datetime	
viewed	tinyint(1)	NN

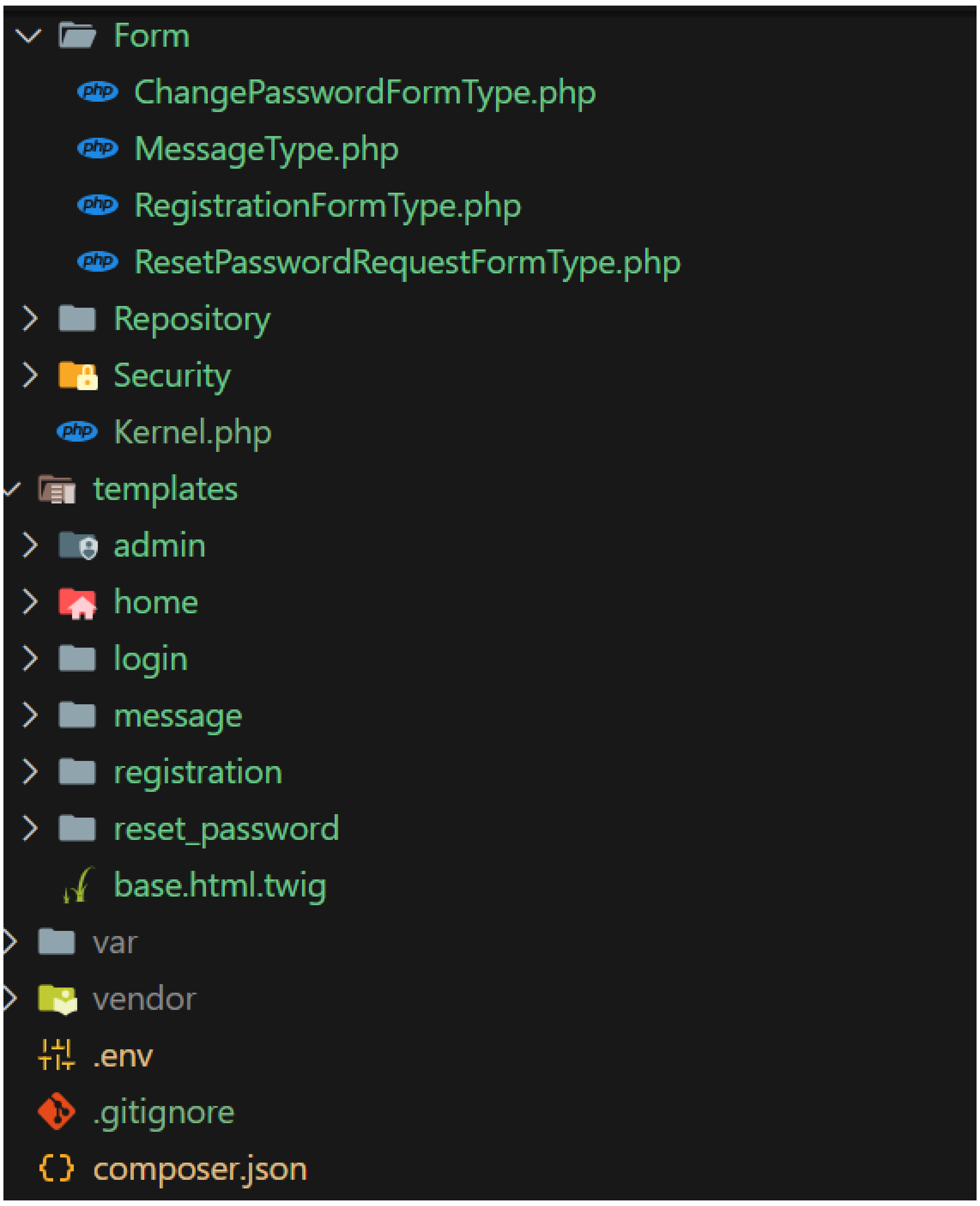
reset_password_request		
id	int	
user_id	int	NN
selector	varchar(20)	NN
hashed_token	varchar(100)	NN
requested_at	datetime	NN
expires_at	datetime	NN

> This Database diagram is generated using dbdiagram.io an online website similar to sql developer.

Summary table



```
symfonyMessagingApp
  bin
  config
  migrations
  public
  src
    Controller
      .gitignore
      AdminController.php
      HomeController.php
      InboxController.php
      LoginController.php
      MessageController.php
      OutboxController.php
      ProfileController.php
      RegistrationController.php
      ResetPasswordController.php
    Entity
      .gitignore
      Message.php
      ResetPasswordRequest.php
      User.php
```



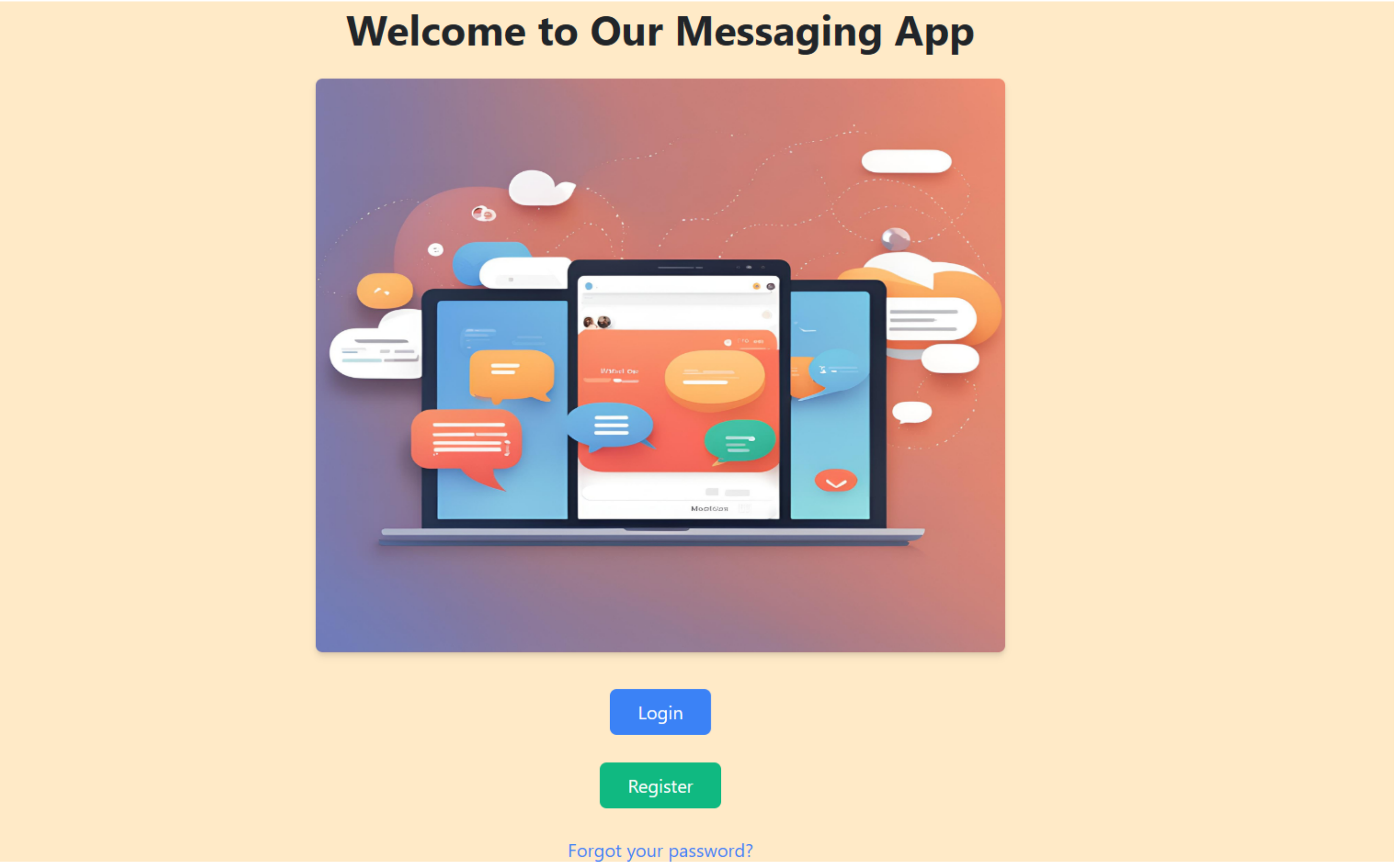
```
  Form
    ChangePasswordFormType.php
    MessageType.php
    RegistrationFormType.php
    ResetPasswordRequestFormType.php
  Repository
  Security
  Kernel.php
templates
  admin
  home
  login
  message
  registration
  reset_password
  base.html.twig
var
vendor
.env
.gitignore
composer.json
```

Files Summary Table

- `bin`: Contains executable binary files.
- `config`: Holds configuration files for the application.
- `migrations`: Contains database migration files.
- `public`: Entry point for the web application (public assets).
- `src`: Source code directory.
- `src/Controller`: Controllers handling HTTP requests.
- `src/Entity`: Database entity classes.
- `src/Form`: Form types for Symfony forms.
- `src/Repository`: Repository classes for database interaction.
- `src/Security`: Security-related components (e.g., user authentication).
- `templates`: Twig templates for rendering HTML.
- `var`: Temporary and cache files.
- `vendor`: Composer dependencies and packages.
- `.env`: Environment configuration file.
- `.gitignore`: Specifies intentionally untracked files to ignore.
- `composer.json`: Defines PHP project dependencies and metadata.
- `composer.lock`: Lock file ensuring consistent dependencies.
- `Kernel.php`: Symfony application kernel.
- `AdminController.php`: Controller managing admin-related actions.

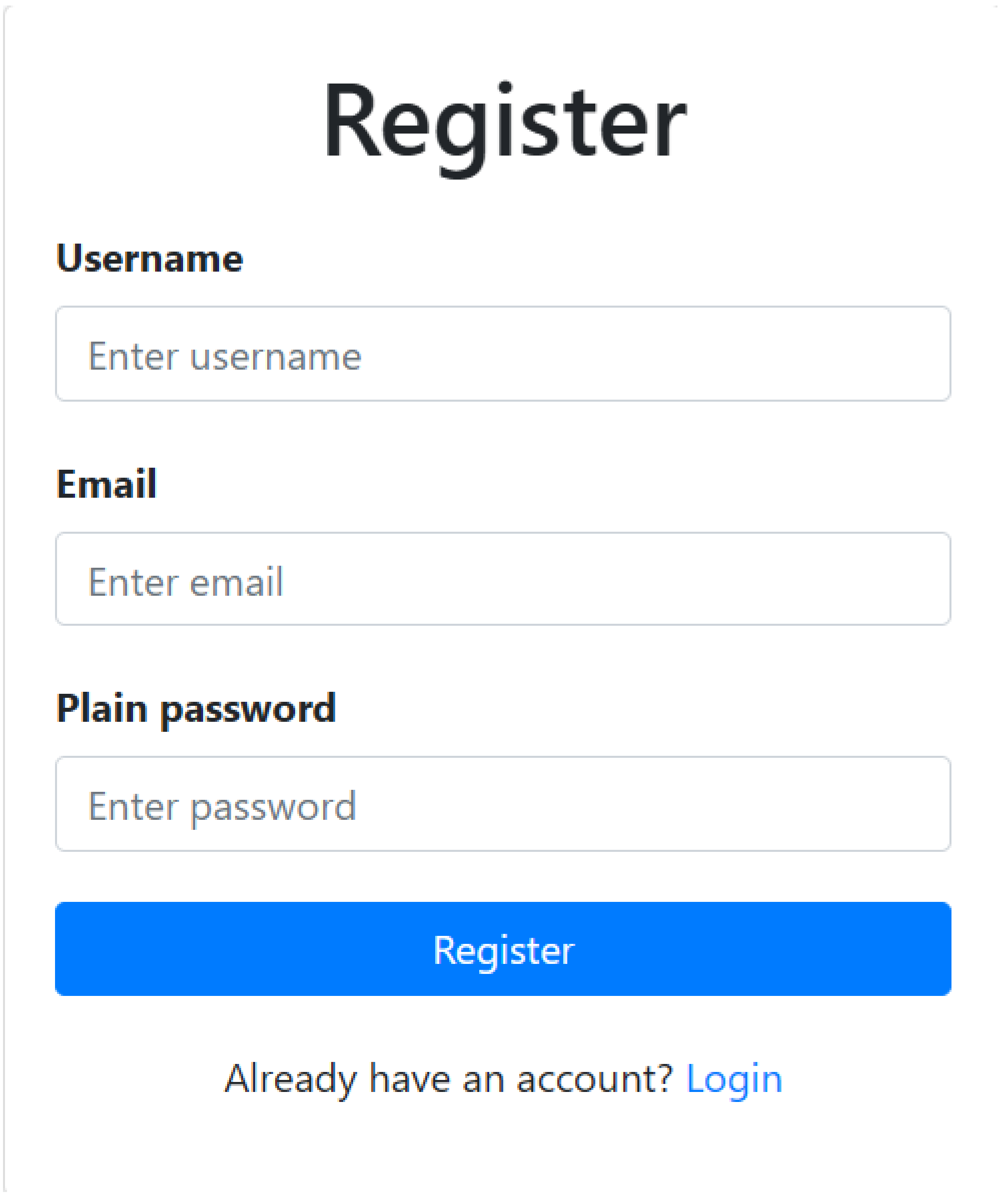
- `HomeController.php`: Controller handling home page actions.
- `InboxController.php`: Controller for handling inbox functionality.
- `LoginController.php`: Controller managing user login.
- `MessageController.php`: Controller handling messaging tasks.
- `OutboxController.php`: Controller for managing outgoing messages.
- `ProfileController.php`: Controller managing user profiles.
- `RegistrationController.php`: Controller handling user registration.
- `ResetPasswordController.php`: Controller for resetting user passwords.
- `Message.php`: Represents a message entity.
- `ResetPasswordRequest.php`: Entity representing a password reset request.
- `User.php`: Represents a user entity.
- `ChangePasswordFormType.php`: Form type for changing passwords.
- `MessageType.php`: Form type related to messaging.
- `RegistrationFormType.php`: Form type for user registration.
- `ResetPasswordRequestFormType.php`: Form type for handling reset password requests.
- `base.html.twig`: Base template for rendering HTML pages.

User Guide

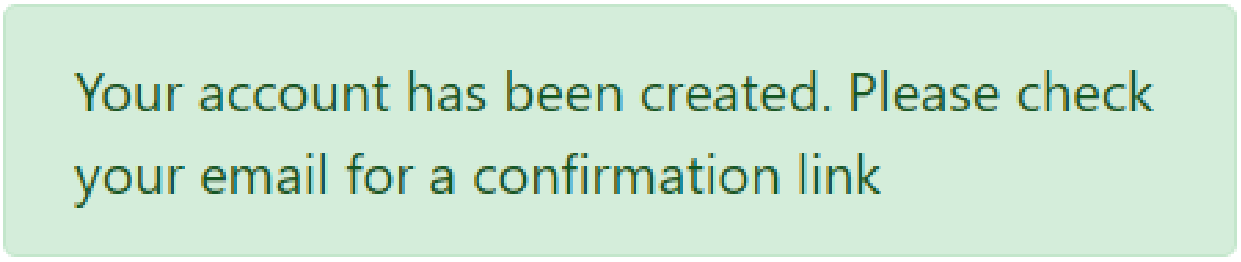


Welcome to Our Messaging App

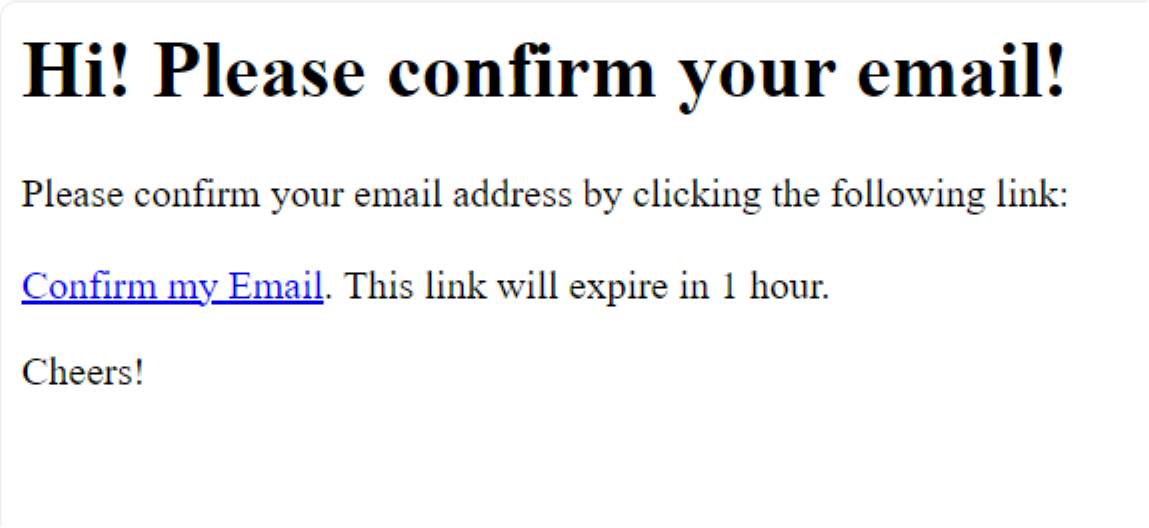
This is the main page index.php or the welcoming page, from here the user can register or login, and if he forgot the password he can click the link ‘`forgot your password?`’.



💡 This is the register.php page, here the user chooses a username, he writes his e-mail and password.


Your account has been created. Please check your email for a confirmation link

💡 Once the registration is completed, the user is asked to check his e-mail for an activation link, he can’t log in without activating his account first.



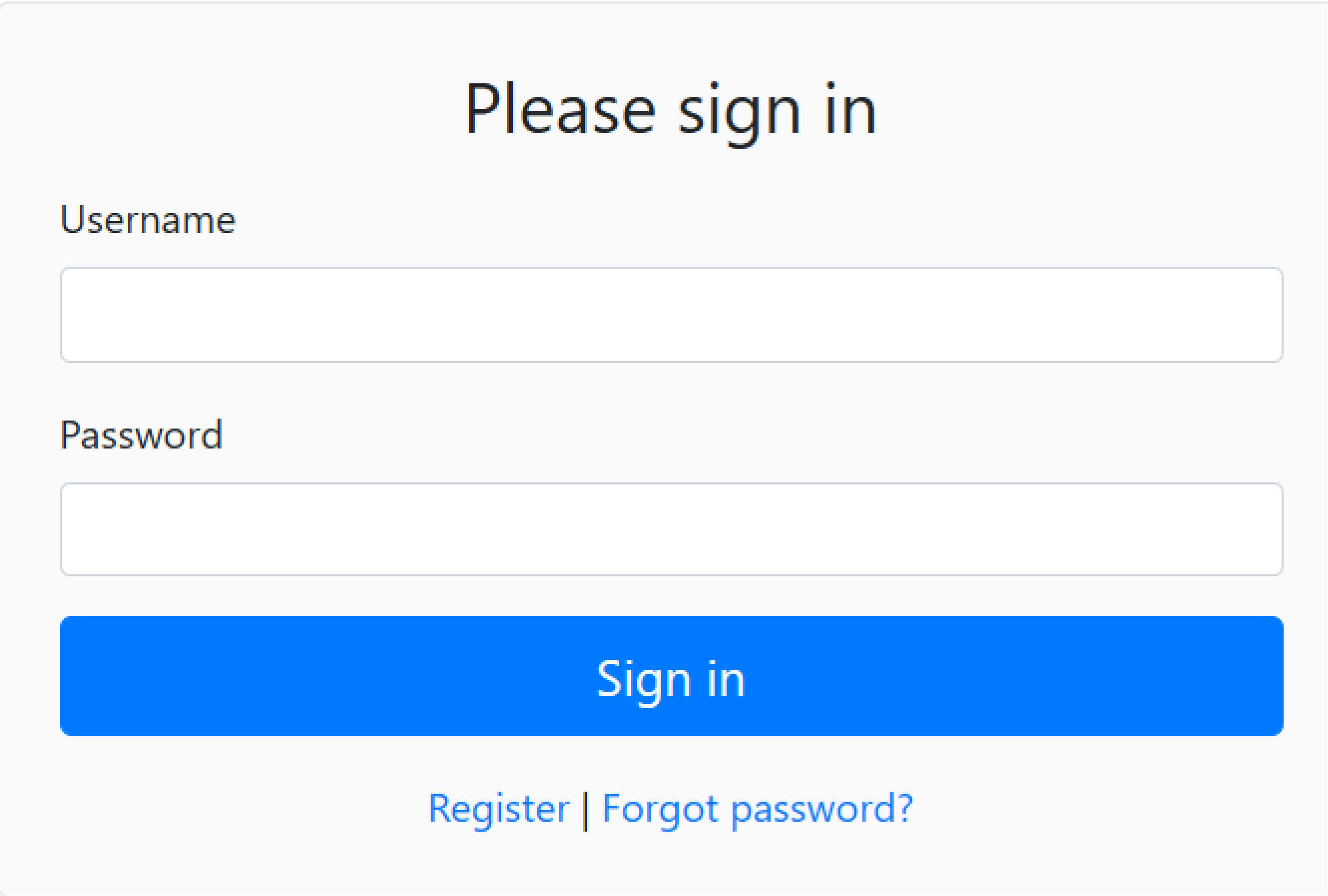
Hi! Please confirm your email!

Please confirm your email address by clicking the following link:

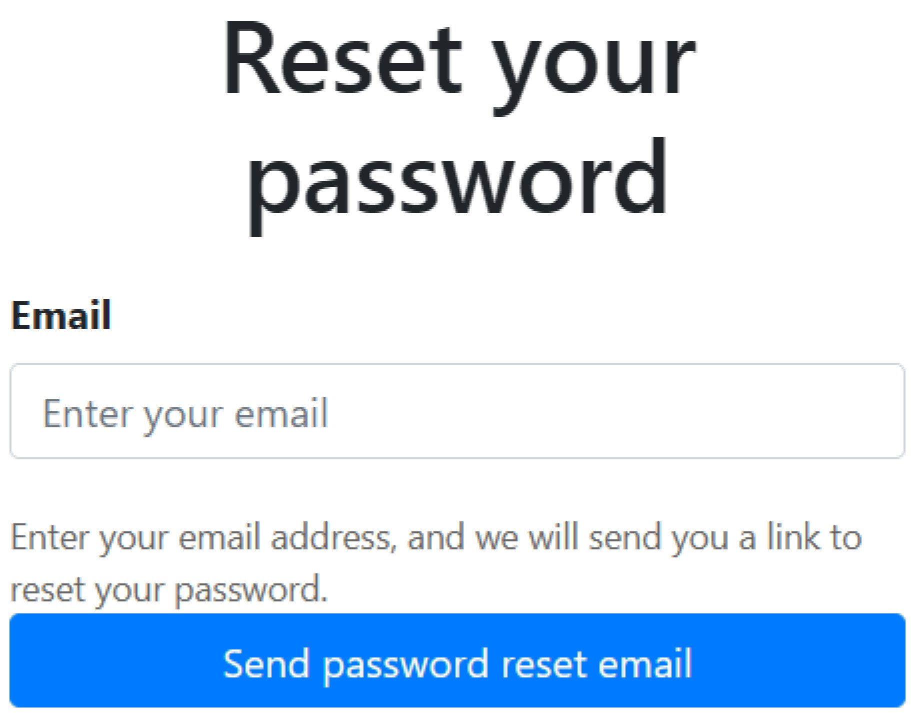
Confirm my Email. This link will expire in 1 hour.

Cheers!

This is how the activation link looks like in the user’s e-mail.



After clicking the activation link, it redirects him to the login page.



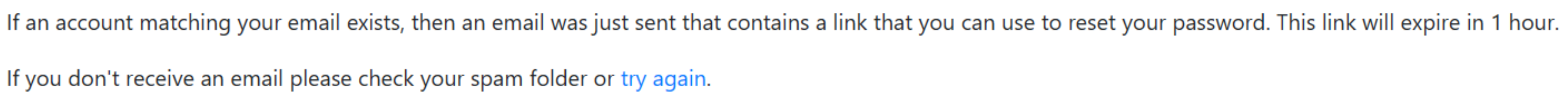
Reset your password

Email

Enter your email

Enter your email address, and we will send you a link to reset your password.

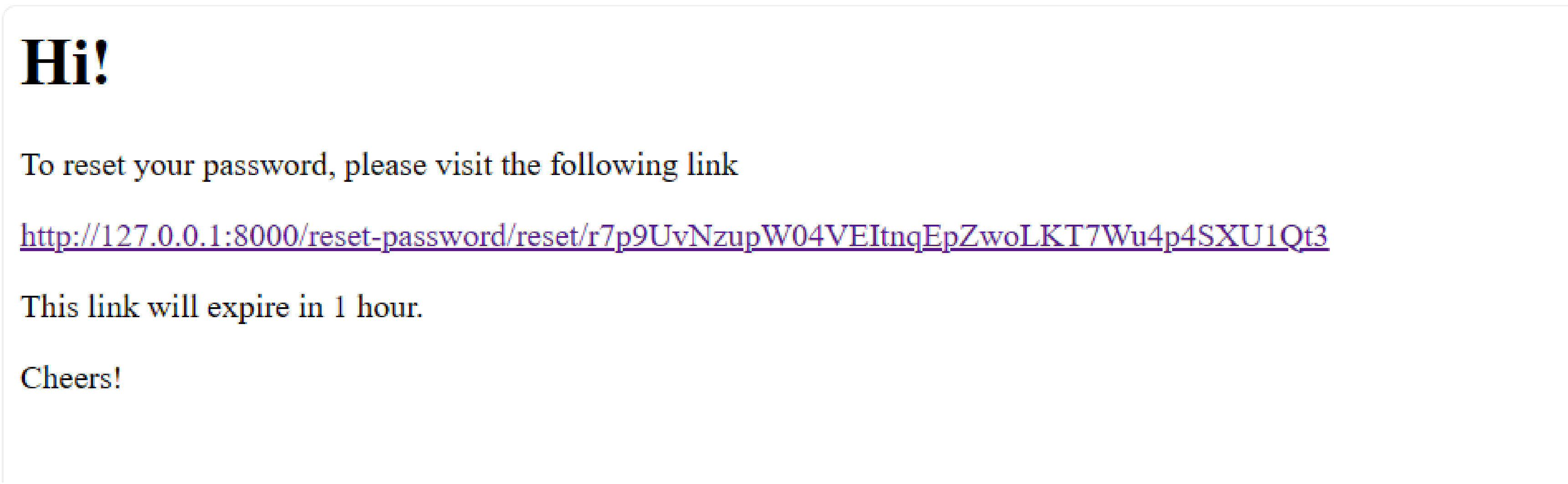
💡 There is also `the forgot your password?` link , in case the user forgot his password, he can click it and he will be asked to enter his email, then he will receive an email to reset his password and make a new one.



If an account matching your email exists, then an email was just sent that contains a link that you can use to reset your password. This link will expire in 1 hour.

If you don't receive an email please check your spam folder or try again.

💡 When the reset is sent, the user will get a message to check his email for a reset password link



Hi!

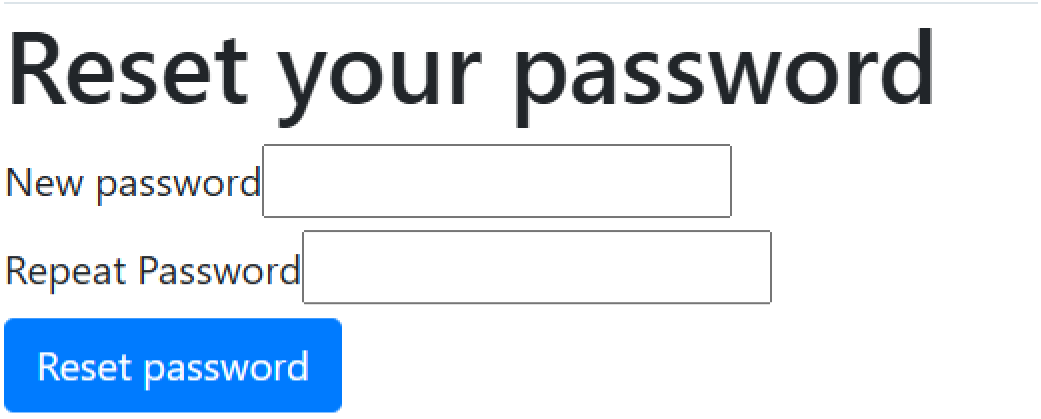
To reset your password, please visit the following link

http://127.0.0.1:8000/reset-password/reset/r7p9UvNzupW04VEItnqEpZwoLKT7Wu4p4SXU1Qt3

This link will expire in 1 hour.

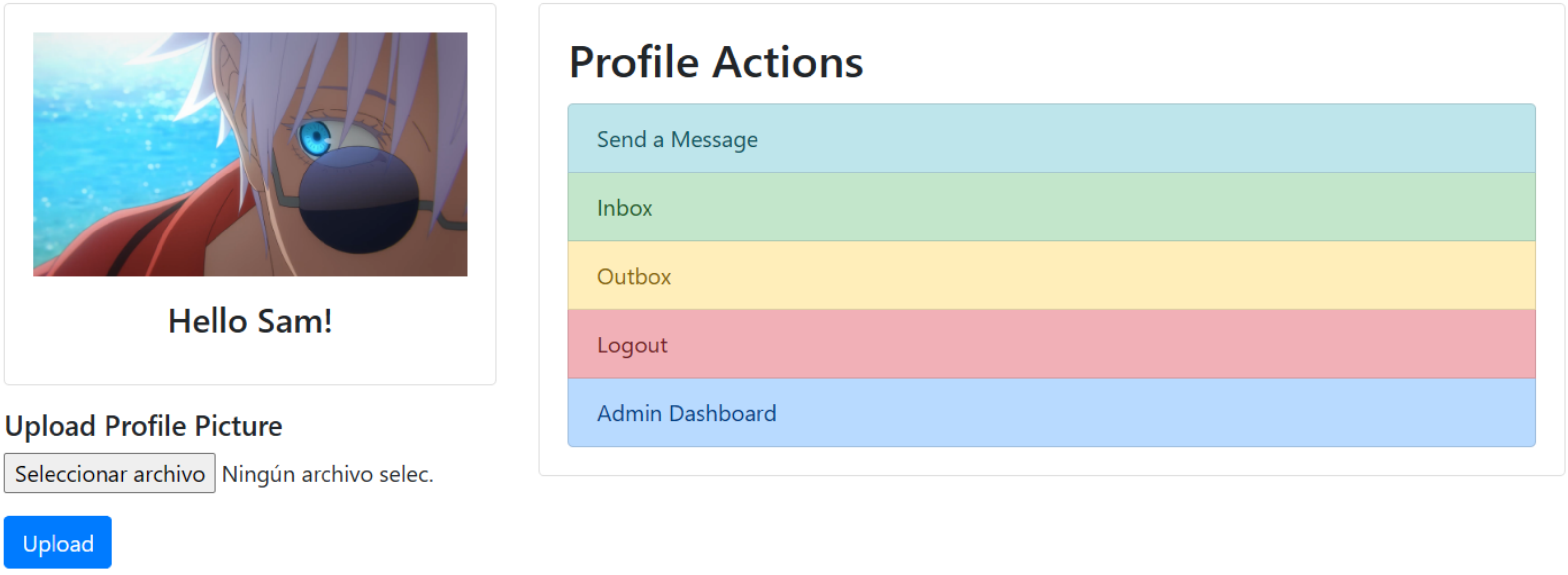
Cheers!

the user will receive an email with a link to reset his password.



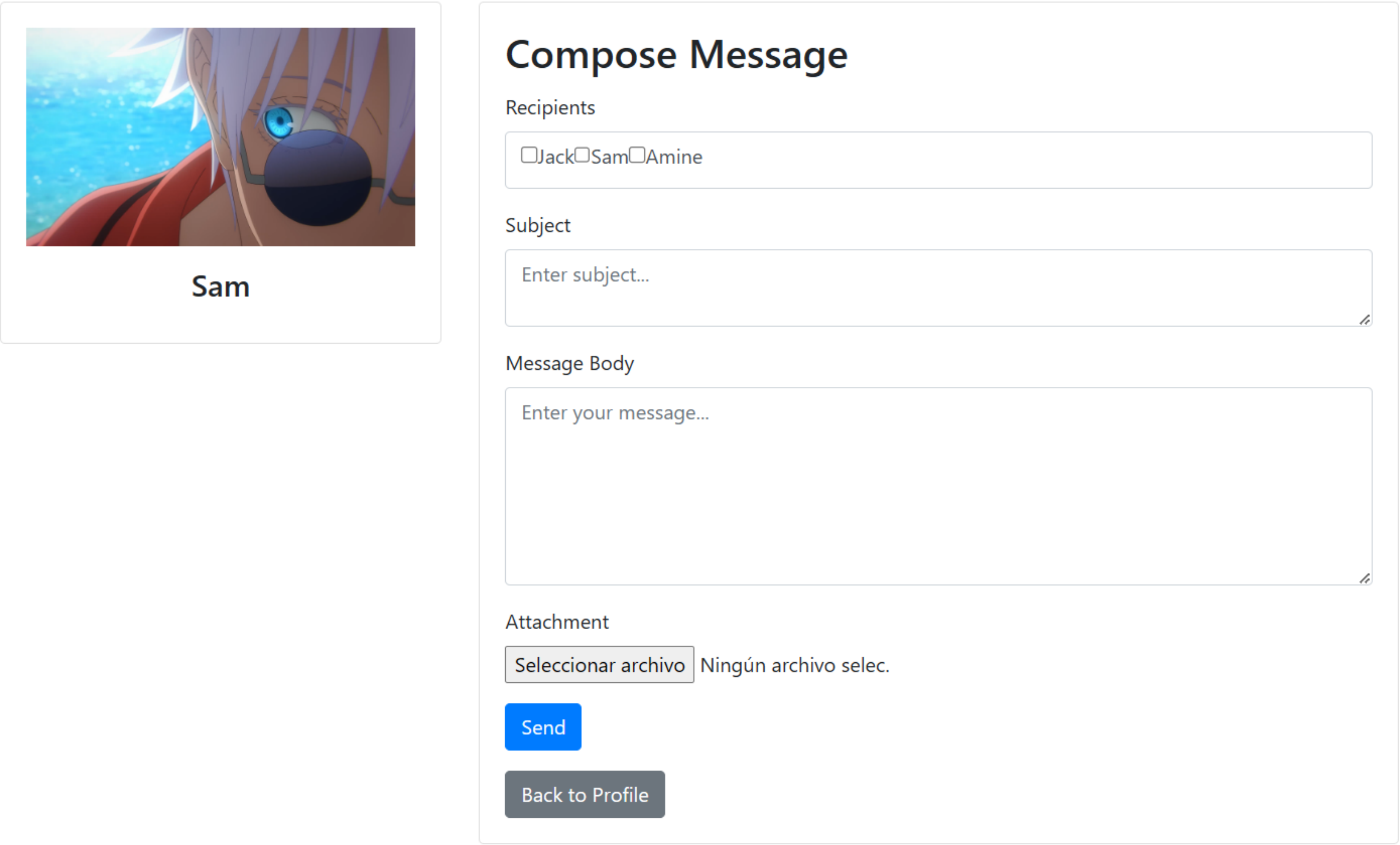
Reset your password

Here the user is asked to introduce a new password and click update password.



Hello Sam!

now after loging-in, the user can access his profile homepage! the user can set a picture to be his user profile picture, and there are profile actions that we will cover next.



Sam

Compose Message

Recipients

□Jack□Sam□Amine

Subject

Enter subject...

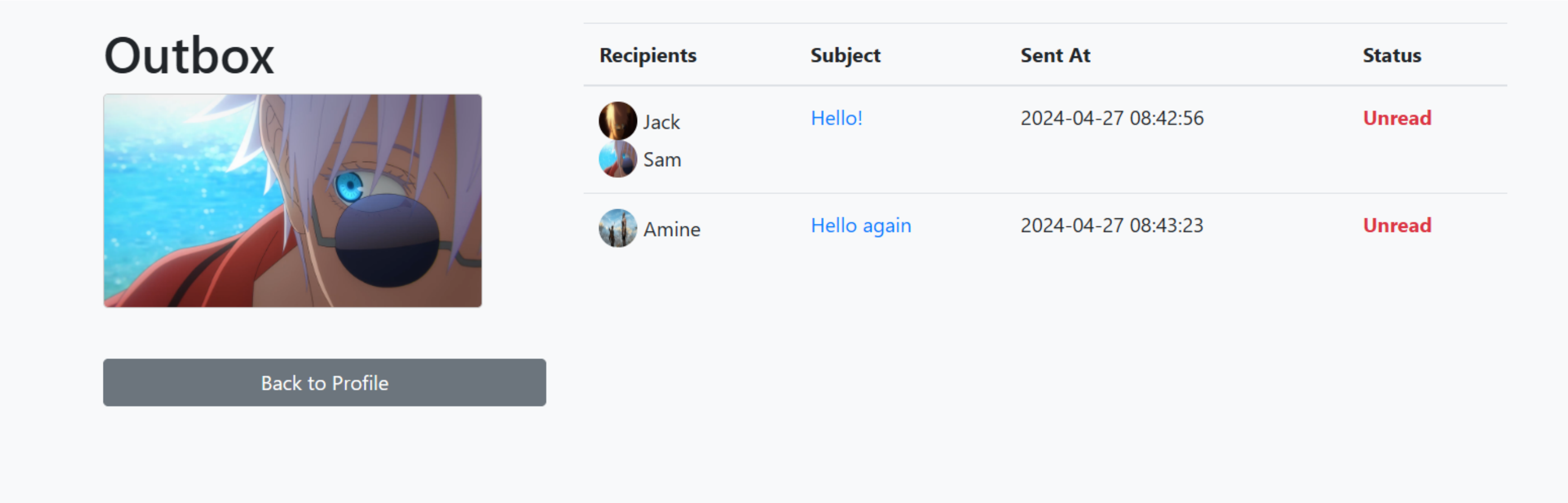
Message Body

Enter your message...

Attachment

Seleccionar archivo Ningún archivo selec.

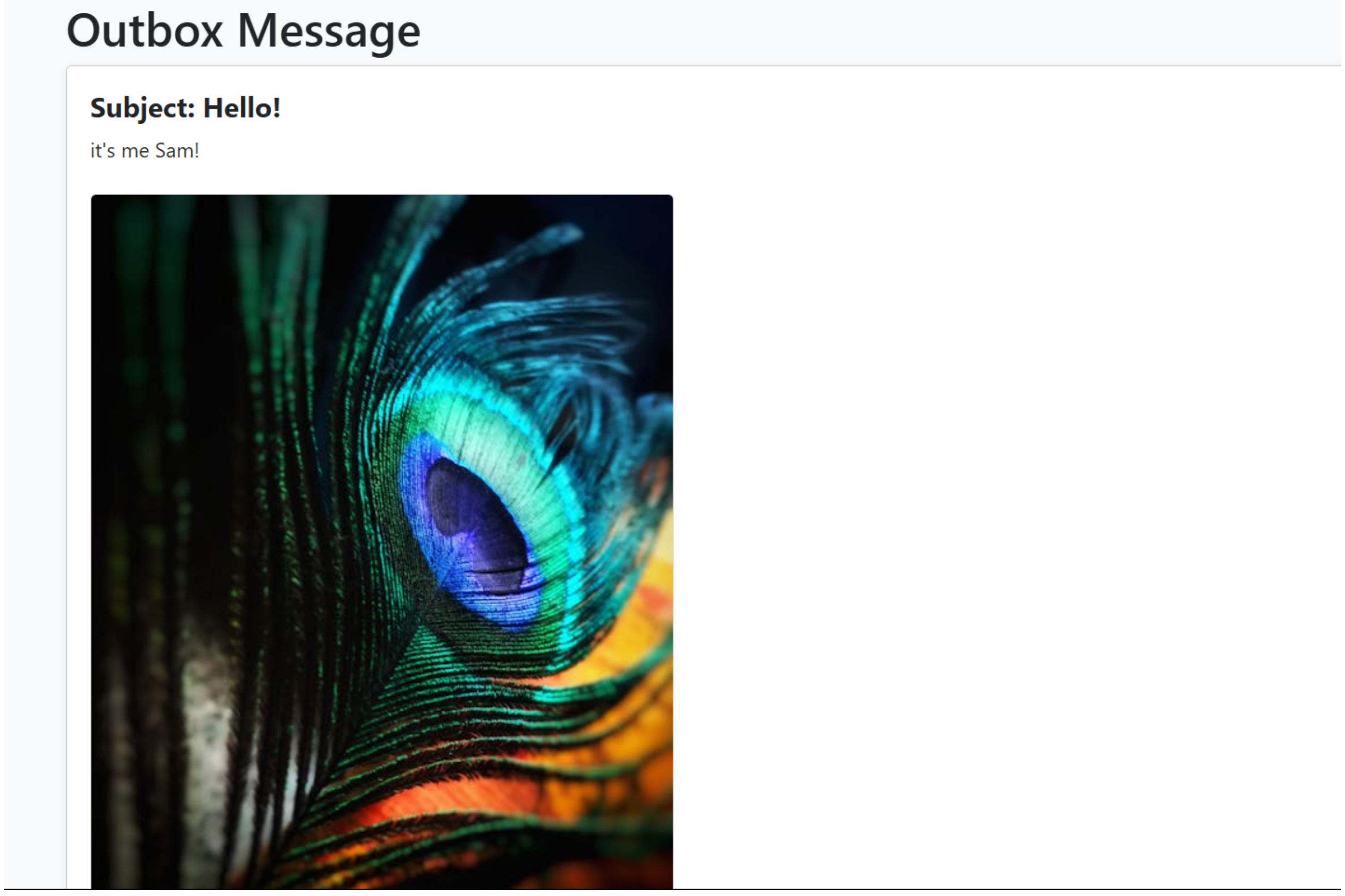
This is the send message page, the user can send a message to one or many recipients at the same time. also he can attach files or images to his message.



Outbox

Recipients	Subject	Sent At	Status
Jack Sam	Hello!	2024-04-27 08:42:56	Unread
Amine	Hello again	2024-04-27 08:43:23	Unread

to check sent messages, the user can access the outbox page. the recipients are shown by name and there profile picture. there is also the subject of the message, the date sent and the status(read/unread).

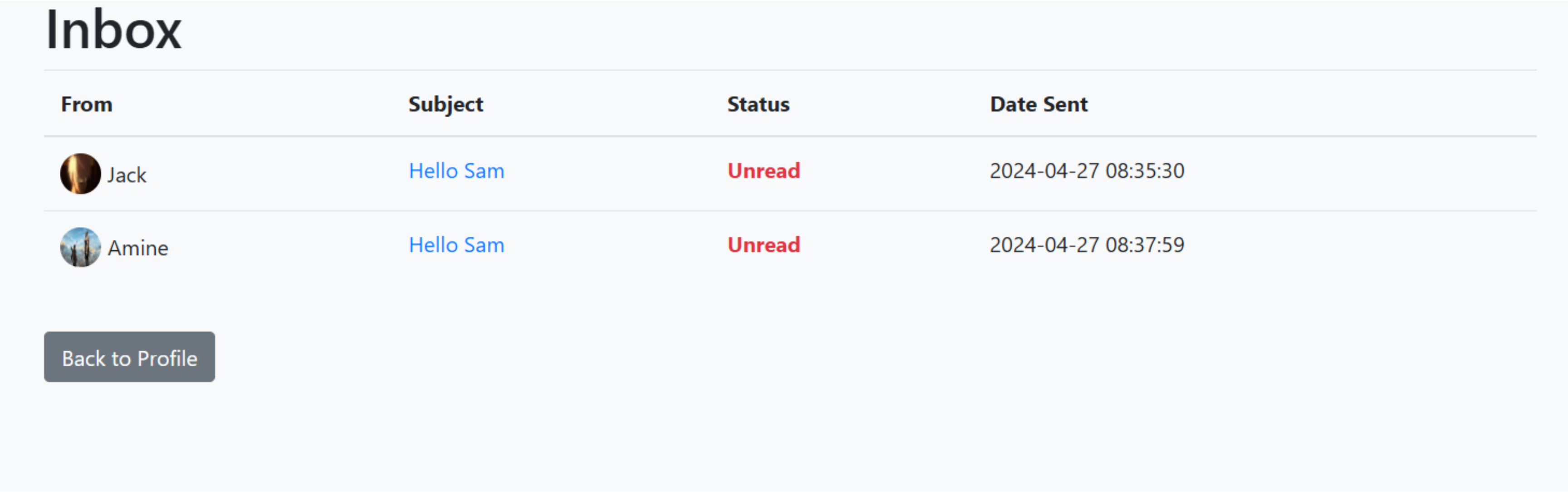


Outbox Message

Subject: Hello!

it's me Sam!

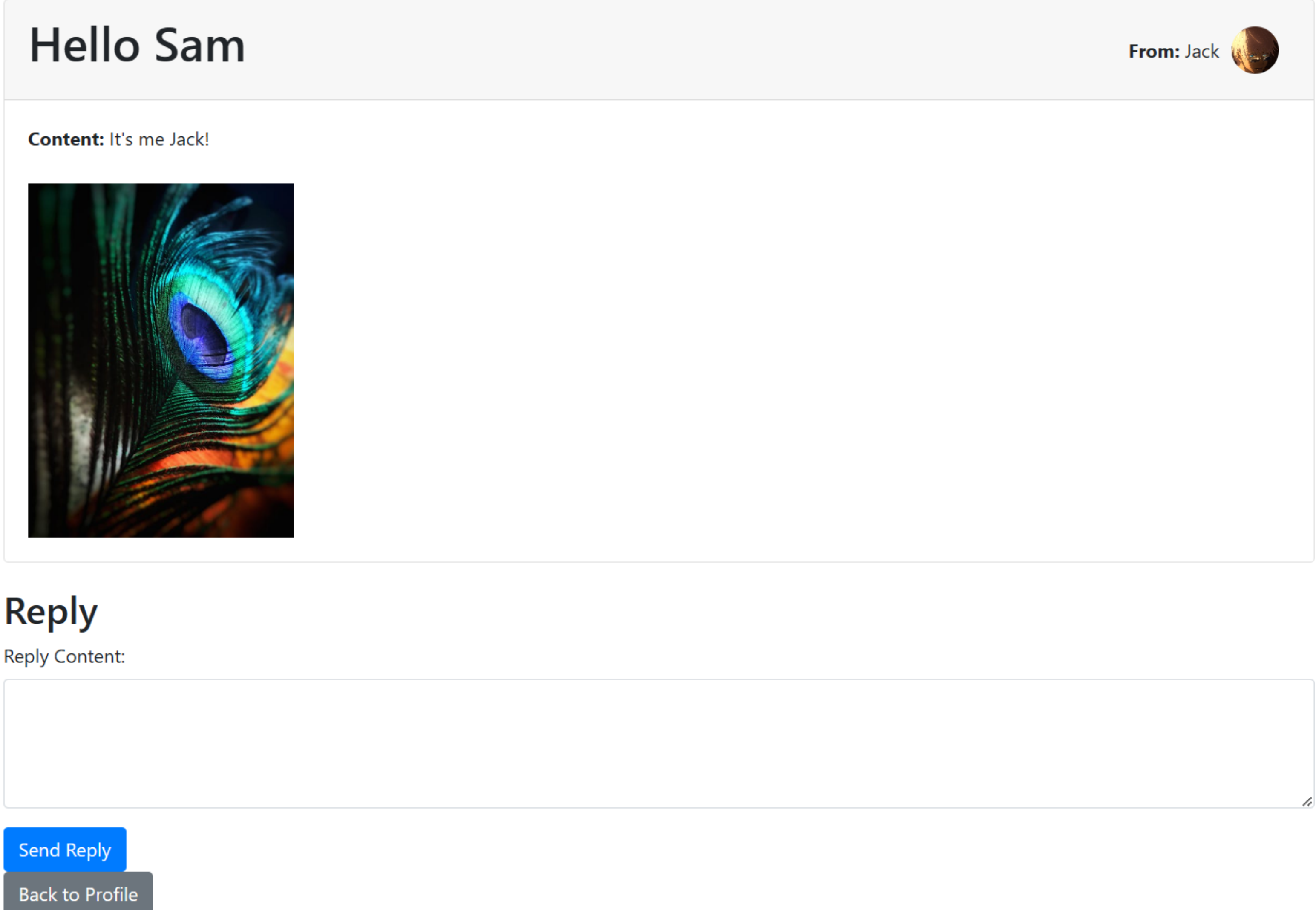
💡 he can see the full content of the message by clicking on the link in the subject column. if the user receives an image as an attachment, it will be directly shown with the message body



Inbox

From	Subject	Status	Date Sent
Jack	Hello Sam	Unread	2024-04-27 08:35:30
Amine	Hello Sam	Unread	2024-04-27 08:37:59

💡 This is the inbox page where the user can see the messages sent to him, with the subject, status, and the time the message was sent.



Hello Sam

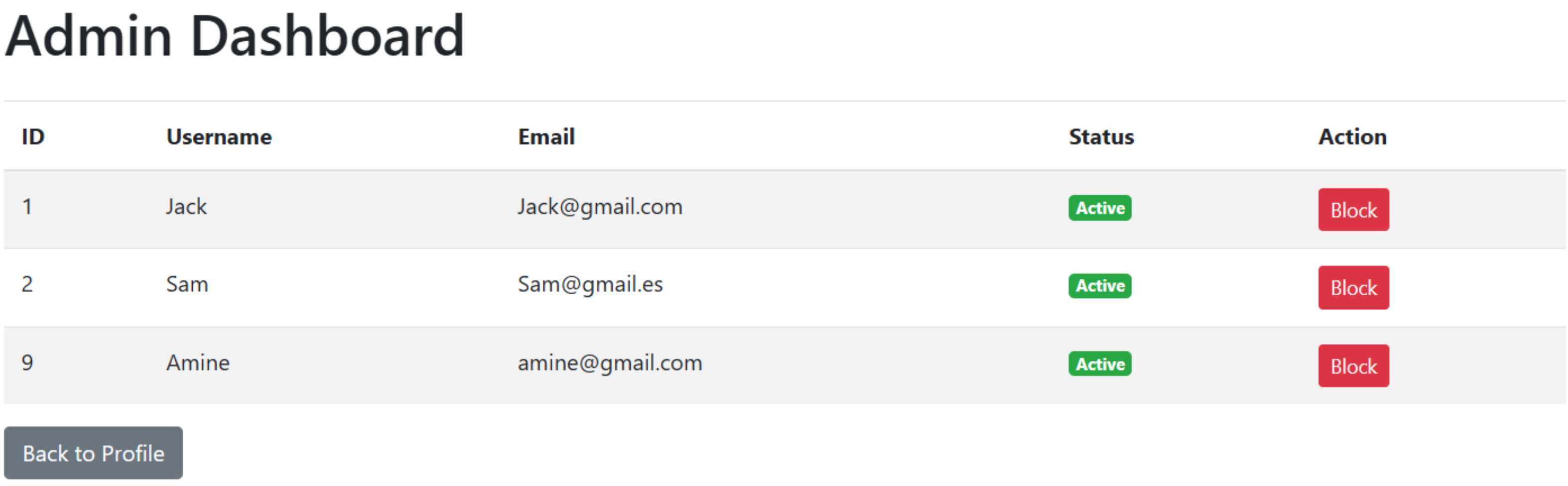
From: Jack

Content: It's me Jack!

Reply

Reply Content:

💡 Same for the inbox you can the click the link in the subject column to see the full content of the message. Alo there is the reply part where you can write your reply.



Admin Dashboard

ID	Username	Email	Status	Action
1	Jack	Jack@gmail.com	Active	Block
2	Sam	Sam@gmail.es	Active	Block
9	Amine	amine@gmail.com	Active	Block

💡 and This is the admin page, in this case Sam the admin has the possibility to block and unblock users, if some user is blocked, he can’t send nore receive messsages, he won’t be able to access his homepage either.

🌟 **Thank You for Reading!** 🌟