Part 2

In this assignment, you will extend your web application developed in Part 1 to include the password reset feature and Single Sign-On (SSO) through Google accounts.

Introduction

You managed to develop everything on time, and the representative from *thin air LTD* was quite satisfied. Still, as every good businessperson, they asked for an additional functionality to allow them to expand their potential customer base even further. This time, they want to implement a Single Sign-On functionality, so that lazy customers are not discouraged by all the typing needed during the registration procedure, and can just click on the Google "Sign In" button.

Moreover, your colleagues were helping you out and they noticed a missing functionality. They pointed out that users often forget their passwords – and now you also have to implement a password recovery function. You know that there are plenty of mechanisms to do so, but your seniors still want to test your capabilities and do not give you any hint on which one would be better: you need to consider each of them and decide base on your assessment. You decide to add the password recovery functionality in the login page.

Task 2.1 Password Reset

Implement a password reset feature for your web application. You are free to choose the best method to achieve this keeping in mind the security aspects. Some options are:

- Resending the same password by email.
- Creating a new password and sending it by email.
- Creating a link with a password reset token.
- Generate a one-time password (OTP), and send it by email.
- ...

You should try to identify **the security risks of each approach**, and choose the one that you think is better. Make sure to consider the **confidentiality** of information, the **integrity** of the whole process, and **actions** you take after a successful password reset.

Note 1. The password reset page must be accessible at `/accounts/password-reset◀`. Users will enter their username to initiate the password reset process. Note that based on your chosen password recovery technique, there will be another set of web pages needed to complete the password reset process.

Note 2. As in the previous assignment, you can just print any information that you are planning to send to the user to the standard output rather than sending an actual email.

Note 3. You are NOT allowed to use the built-in password reset features of the framework (e.g., the default password reset web pages of Django), but rather you should try to implement your own. Specifically, the controller functions that handle the HTTP requests must be developed by you for the password recovery.

Task 2.2 Single Sign-On

Next, we will integrate a Google SSO feature for our ecommerce web application leveraging the Google Identify Platform. Your web site (i.e., the SP) will authenticate the end users by Google accounts (i.e., the IdP) leveraging **the client-side JavaScript code**.

Note. In this assignment, we will use the OpenID Connect (OIDC), which is an *authentication* layer on the top of the oAuth 2.0 *authorization* framework. This authentication flow results in an OIDC *ID Token*. An *ID Token* is a Json Web Token (JWT) that contains user's identity data and comprises three parts: a header, a body and a signature, as specified in IETF RFC 7519.

Creating Authorization Credentials. To use OAuth 2.0 for accessing Google APIs, your application must have authorization credentials that identify it to Google's OAuth 2.0 server. To create a *client ID*, follow the steps below:

1. Visit the Developer's Credentials page: `https://console.developers.google.com/apis/credentials`.
2. Click Create credentials > OAuth client ID.

Next, specify your app's client ID leveraging the `google-signin-client_id` meta element in the HTML of your login page. For more details, see `https://developers.google.com/identity/sign-in/web/sign-in`.

Expected Behaviour. At a high level, your authentication flow should match the following specification:

1. Include a IdP Sign-In button in your login page at `/accounts/login`.
2. When the user clicks on the IdP Sign-In button, a new browser tab will open by the SP's client-side code where the user is prompted to follow the authentication procedure of the IdP, if not already authenticated.
3. After successful authentication at IdP, the IdP returns to the SP's client-side code an *ID Token* containing the identity data of the user.
4. The SP will process the *ID Token* and authenticate the user accordingly.

For implementation details, follow the steps in the Google Sign-In documentation available at `https://developers.google.com/identity/sign-in/web/sign-in`.