Design Document

Team members :

Abhinav Pratap Singh(180010001)
B Sai Yashwanth(180010010)
Tarun Goyal(180010038)
Utkarsh Prakash(180030042)

High-Level Design Document

Description:

This document gives a high-level design description of integrating Google Login with a Django web application. Google uses OAuth 2.0 for authentication and authorization purposes. In this document we provide the structure of the system such as the application architecture, application flow and database design.

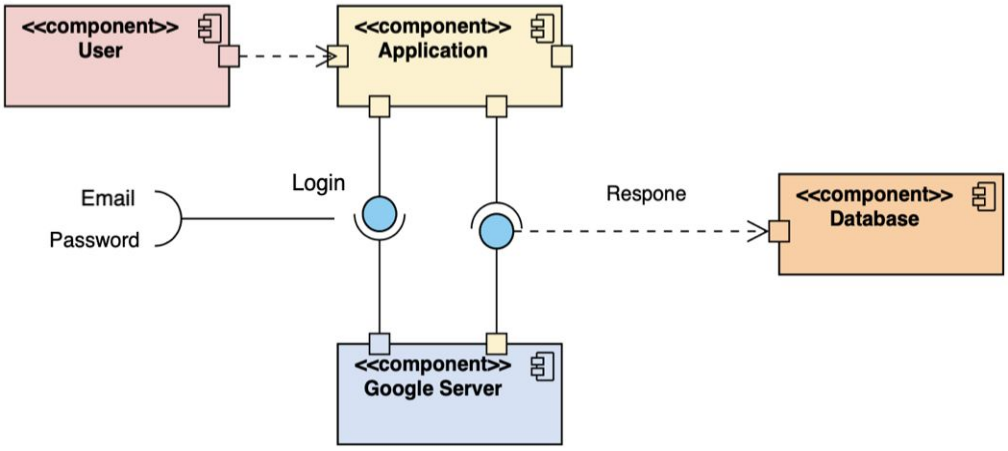
Tools Used:

- Django
- Django-allauth library
- HTML
- SQLite Relational Database
- Google API console

Interfaces:

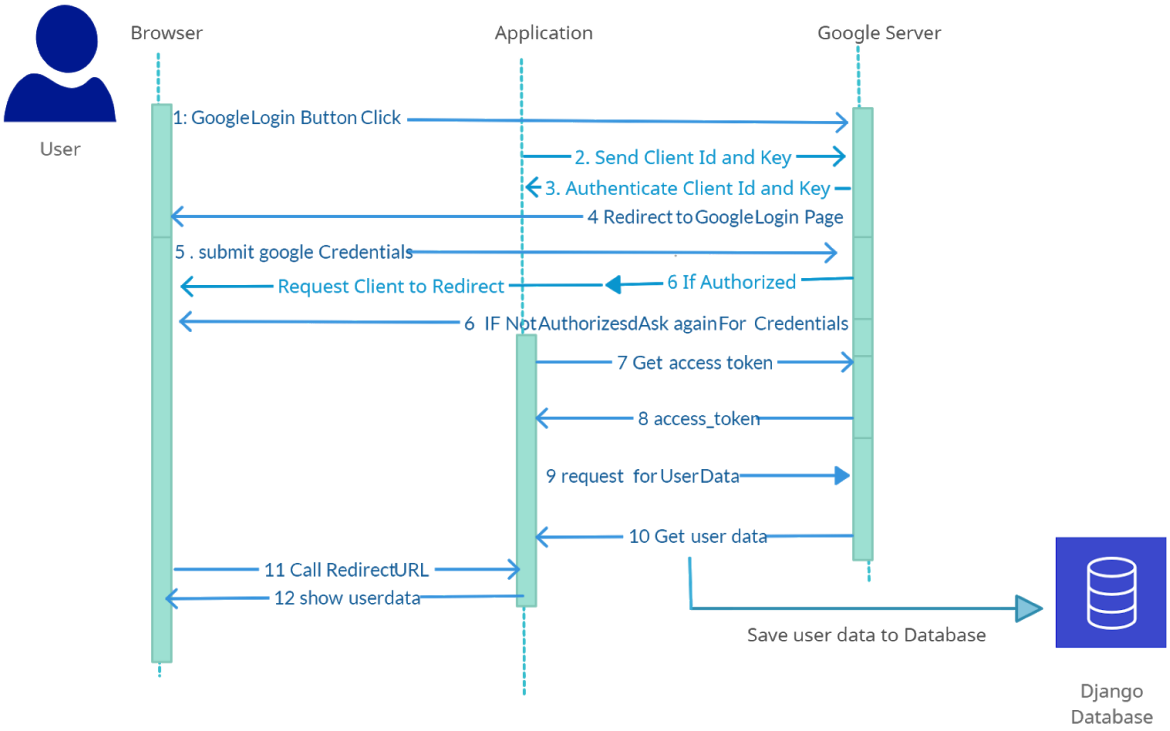
- **Application Login Page login.html:** This page will give the option whether the user wants to login from Google or Facebook or normal login.
- **Google Login Page:** This page will ask for the google credentials such as email ID and password.
- **Search Page base.html:** This is the page to which the user will be redirected to when he/she successfully login. This page provides the option to search media.

Component Diagram :



User uses application. Application provides the interface for login. Login Interface also requires Email , Password. This request will go to google server, and then google server gives token and user data in response which is further stored into the database. After getting response from google server, application will redirect user to home page.

Application Flow (Sequence Diagram):



User clicks Google Login Button, it accesses the google server through OAuth Client ID and Secret Key for authentication and authorization. After successful authentication browser gets redirected to google login page. In this page the user Submits Google Login Credentials such as their Google account email ID and password. If the email and password are correct, then google server executes the call back. If email and password are incorrect then it will ask again for correct credentials. The application will ask for access token and user data. After getting the user data and access token, the callback function stores the data in the Django SQLite database. The application will show the home page (search page) with the user data.

Database Design:

Django Administration

E-mail addresses : E-mail address, user, primary, verified

Users : storing username, email address, first name, last name, staff status(tick for superusers)

Sites : Domain name, Display name

Social Accounts : User, UID, provider

Social applications tokens : app, account, token, expires at

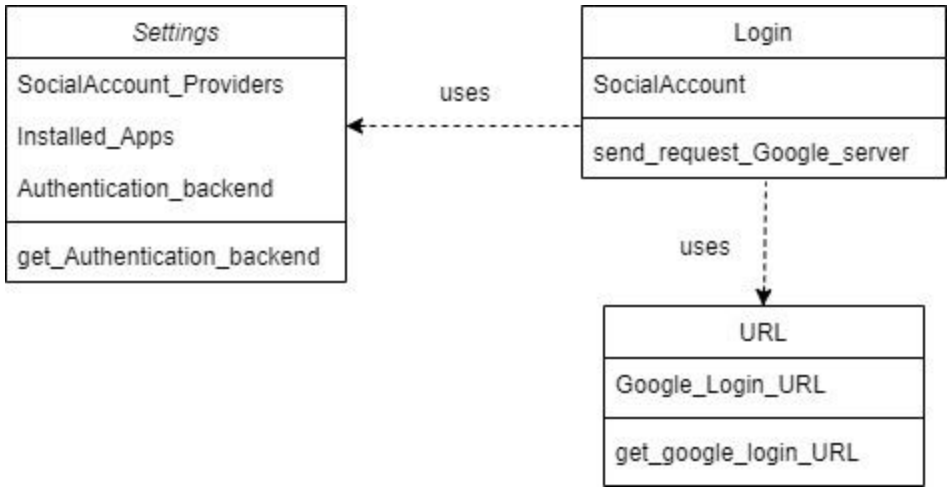
Social Applications : Name(GOOGLE API) , Provider

Portability

The application is compatible with Windows, Ubuntu and MacOSX. It can be run on any machine which has python3.6 (or later), Django (2.2.4), django-allauth (0.44.0) and django-google-api (1.0.1) installed.

Low-Level Design Document

Class Diagram



Here the Login uses the Settings to get the authentication backend information. It also uses the URL to get the URL of the google login server. Then it sends the OAuth Client ID and Secret Key for authentication and authorization. The required client ID and secrets for interacting with Google are configured in the database via the Django admin using the SocialApp model. Once the authentication is successful then the user is asked to fill the email credentials. Then again a request is sent to the Google server for verifying the email and the password. After a successful validation, google server sends the access token and the user data associated with the corresponding Google account. The callback function will store this data in the database. This callback function is implemented in the Login.

Interfaces

Interface	**Description**
Application Login	This page will give the option whether the user wants to login from Google or Facebook or normal login. If a user clicks on Google Login, he/she will be redirected to google login page.
Google Login	This page will ask for the google credentials such as email ID and password. If credentials filled by the user are correct then the user is redirected to the search page otherwise the application will ask the credentials again.

Search Page	This is the page to which the user will be redirected to when he/she successfully login. This page provides the option to search media (Google Drive, Youtube, Database).

Security

While creating the OAuth Client ID and secret key we specify the origin URL and redirect URL, so if we try to access the google server from a different URL we will get an error. Moreover, in the Settings we provide scope which restricts the information associated with the Google Account that is retrieved from the Google Server. The access token of a particular account has a certain TTL (Time to Live), after which it expires and cannot be used again. In this way we ensure that the user data is secured.