PROJECT REPORT
ON
"LIVE CHAT WEBSITE USING PYTHON - DJANGO FRAMEWORK "

Submitted in partial fulfillment of
The requirements for the award of the Degree of

BACHELOR OF
COMPUTER APPLICATION

Submitted By:

HARDIK MOHNANI

Under the esteemed guidance of :

PROF. BHAVIK PATHAK

FROM : SHRI. K.M. & SMT. K.K. SAVJANI BBA/BCA COLLEGE
VERAVAL, GUJARAT-362265

Submitted To : BKNMU

Year of Submission : 2022-2023

Bachelor of Computer Application(SEM-6)

BHAKTA KAVI NARSINH MEHTA. UNIVERSITY. JUNAGADH– INDIA

☯ INDEX ☯

1	Acknowledgement	5
2	Preface	6
3	Project Profile	7
4	Development Tools	8
5	Django + Python Hardware & Software Requirement	19
6	SDLC	20
7	System Analysis	22
8	About Project	26
9	Dataflow Diagram	28
10	Database (Table Layout)	32
11	File manager	41

12	CMD Command	43
13	Screen Shots With Details	44
15	Coding (Implementation)	55
16	Testing	102
17	Future Enhancement	105
18	Bibliography	107

☯ ACKNOWLEDGEMENT ☯

Our Website, which you are using, is the result of many people's dedication. It is the cumulative efforts of many minds working together day and night that gave us the contentment of developing the website.

Special thanks to Prof. (Dr) Jigar Raval Sir, The Principal for his great support. We express our gratitude to for guidance and who kept the things on track and also to all other faculty members who helped us directly or indirectly.

I would like to express our deepest sense of gratitude to our project guides Prof. Bhavik Pathak Sir who have always inspired me and have directed towards us successful completion of our project. They guide throughout the project and their encouragement has left us indebted to them.

THANKS TO ALL. GREAT JOB!!

☯ PREFACE ☯

The practical training is almost important in understanding the theoretical aspect. Viewing to this importance I have prepaid this project report to enrich my knowledge regarding Web Development.

This project is in partial fulfillment of BCA 6th semester. It used python Django. This is very popular framework in the market. The Website records details pertaining to the Customer, who open the Website, and open the Homepage.

This Document also contains a brief profile of the description of the project, data flow diagram. Finally, some screen layouts.

☯ Project Profile ☯

Python Version	3.11.0
Django Version	4.1
Site Language	English _US
Database	Db.sqlite3
Database size	224 KB
Total installation size	13.6 MB
Web server	HTTP Server
Languages	DJANGO(PYTHON), CSS ,HTML , BOOTSTRAP, JAVASCRIPT
Guide	Prof. Bhavik Pathak
Documentation tool	Microsoft Word
Operating System :	Windows 11
Submitted By	Hardik Mohnani

☯ Development Tools ☯

➢ ABOUT PYTHON :

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- System scripting.
- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python can be used for rapid prototyping, or for production-ready software development.

➢ ABOUT Django (Python):

Django is a Python framework that makes it easier to create web sites using Python.

Django takes care of the difficult stuff so that you can concentrate on building your web applications.

Django emphasizes reusability of components, also referred to as DRY (Don't Repeat Yourself), and comes with ready-to-use features like login system, database connection and CRUD operations (Create Read Update Delete).

Django follows the MVT design pattern (Model View Template).

- Model - The data you want to present, usually data from a database.
- View - A request handler that returns the relevant template and content - based on the request from the user.

- **Template - A text file (like an HTML file) containing the layout of the web page, with logic on how to display the data.**

->Model

The model provides data from the database.

In Django, the data is delivered as an Object Relational Mapping (ORM), which is a technique designed to make it easier to work with databases.

The most common way to extract data from a database is SQL. One problem with SQL is that you have to have a pretty good understanding of the database structure to be able to work with it.

Django, with ORM, makes it easier to communicate with the database, without having to write complex SQL statements.

The models are usually located in a file called models.py

->View

A view is a function or method that takes http requests as arguments, imports the relevant model(s), and finds out what data to send to the template, and returns the final result.

The views are usually located in a file called views.py.

->**<u>Template</u>**

A template is a file where you describe how the result should be represented.

Templates are often .html files, with HTML code describing the layout of a web page, but it can also be in other file formats to present other results, but we will concentrate on .html files.

Django uses standard HTML to describe the layout, but uses Django tags to add logic:

The templates of an application is located in a folder named templates

->**<u>URLs</u>**

Django also provides a way to navigate around the different pages in a website.

When a user requests a URL, Django decides which *view* it will send it to.

This is done in a file called urls.py.

->**<u>So, What is Going On?</u>**

When you have installed Django and created your first Django web application, and the browser requests the URL, this is basically what happens:

1. **Django receives the URL, checks the urls.py file, and calls the view that matches the URL.**
2. **The view, located in views.py, checks for relevant models.**
3. **The models are imported from the models.py file.**
4. **The view then sends the data to a specified template in the template folder.**
5. **The template contains HTML and Django tags, and with the data it returns finished HTML content back to the browser.**

Django can do a lot more than this, but this is basically what you will learn in this tutorial, and are the basic steps in a simple web application made with Django

->Django History

Django was invented by Lawrence Journal-World in 2003, to meet the short deadlines in the newspaper and at the same time meeting the demands of experienced web developers.

Initial release to the public was in July 2005.

Latest version of Django is 4.0.3 (March 2022).

➢ <u>**ABOUT DATABASE (MYSQLITE3):**</u>

SQLite is a software library that provides a relational database management system. The lite in SQLite means lightweight in terms of setup, database administration, and required resources.

SQLite has the following noticeable features: self-contained, server less, zero-configuration, transactional.

Server less

Normally, an RDBMS such as MySQL, PostgreSQL, etc., requires a separate server process to operate. The applications that want to access the database server use TCP/IP protocol to send and receive requests. This is called client/server architecture.

SQLite uses dynamic types for tables. It means you can store any value in any column, regardless of the data type.

->Database setup

Now, open up mysite/settings.py. It’s a normal Python module with module-level variables representing Django settings.

SQLite is included in Python, so you won’t need to install anything else to support your database. When starting your first real project, however, you may want to use a more scalable database like PostgreSQL, to avoid database-switching headaches down the road.

If you wish to use another database, install the appropriate database bindings and change the following keys in the DATABASES 'default' item to match your database connection settings:

- **ENGINE – Either 'django.db.backends.sqlite3', 'django.db.backends. postgresql', 'django.db.backends.mysql', or 'django.db.backends.oracle'. Other backends are also available.**
- **NAME – The name of your database. If you’re using SQLite, the database will be a file on your computer; in that case, NAME should be the full absolute path, including filename, of that file. The default value, BASE_DIR / 'db.sqlite3', will store the file in your project directory.**

If you are not using SQLite as your database, additional settings such as USER, PASSWORD, and HOST must be added. For more details, see the reference documentation for DATABASES.

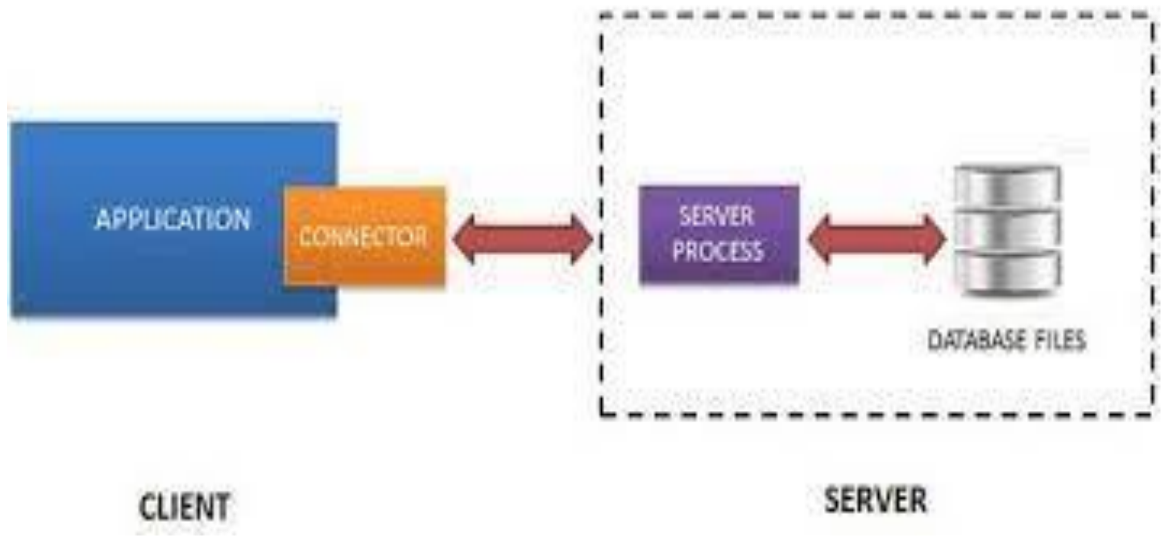
While you're editing mysite/settings.py, set TIME_ZONE to your time zone.

Also, note the INSTALLED_APPS setting at the top of the file. That holds the names of all Django applications that are activated in this Django instance. Apps can be used in multiple projects, and you can package and distribute them for use by others in their projects.

By default, INSTALLED_APPS contains the following apps, all of which come with Django:

- **django.contrib.admin – The admin site. You'll use it shortly.**
- **django.contrib.auth – An authentication system.**
- **django.contrib.contenttypes – A framework for content types.**
- **django.contrib.sessions – A session framework.**
- **django.contrib.messages – A messaging framework.**
- **django.contrib.staticfiles – A framework for managing static files.**

➢ How Does MySQLLITE Database Work:



sqlite3' in your project directory. The file is a database file that will keep all of the data that you will be generating. Since Django is a server-side framework, it treats your computer as the host when you run the server from the command line or terminal.

Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases. There is no need to install this module separately as it comes along with Python after the 4.1 version.

➢ ABOUT HTML:

HTML is at the core of every web page, regardless the complexity of a site or number of technologies involved. It's an essential skill for any web professional. It's the starting point for anyone learning how to create content for the web. And, luckily for us, it's surprisingly easy to learn.

<p>This is a paragraph.</p>

Using HTML, you can add headings, format paragraphs, control line breaks, make lists, emphasize text, create special characters, insert images, create links, build tables, control some styling, and much more.

➢ ABOUT CSS:

CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page.

Whereas HTML was the basic structure of your website, CSS is what gives your entire website its style. Those slick colors, interesting fonts, and background images? All thanks to CSS. This language affects the entire mood and tone of a web page, making it an incredibly powerful tool -- and an important skill for web developers to learn. It's also what allows websites to adapt to different screen sizes and device types.

➢ ABOUT JAVASCRIPT :

JavaScript is a more complicated language than HTML or CSS, and it wasn't released in beta form until 1995. Nowadays, JavaScript is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality.

In short, JavaScript is a programming language that lets web developers design interactive sites. Most of the dynamic behavior you'll see on a web page is thanks to JavaScript, which augments a browser's default controls and behaviors.

➢ <u>ABOUT BOOTSTRAP :</u>

- **Bootstrap is the most popular HTML, CSS and JavaScript framework for developing a responsive and mobile friendly website.**
- **It is absolutely free to download and use.**
- **It is a front-end framework used for easier and faster web development.**
- **It includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many others.**
- **It can also use JavaScript plug-ins.**
- **It facilitates you to create responsive designs.**

☯ DJANGO+PYTHON HARDWARE & SOFTWARE REQUIREMNT ☯

❖ SOFTWARE REQUIREMENT

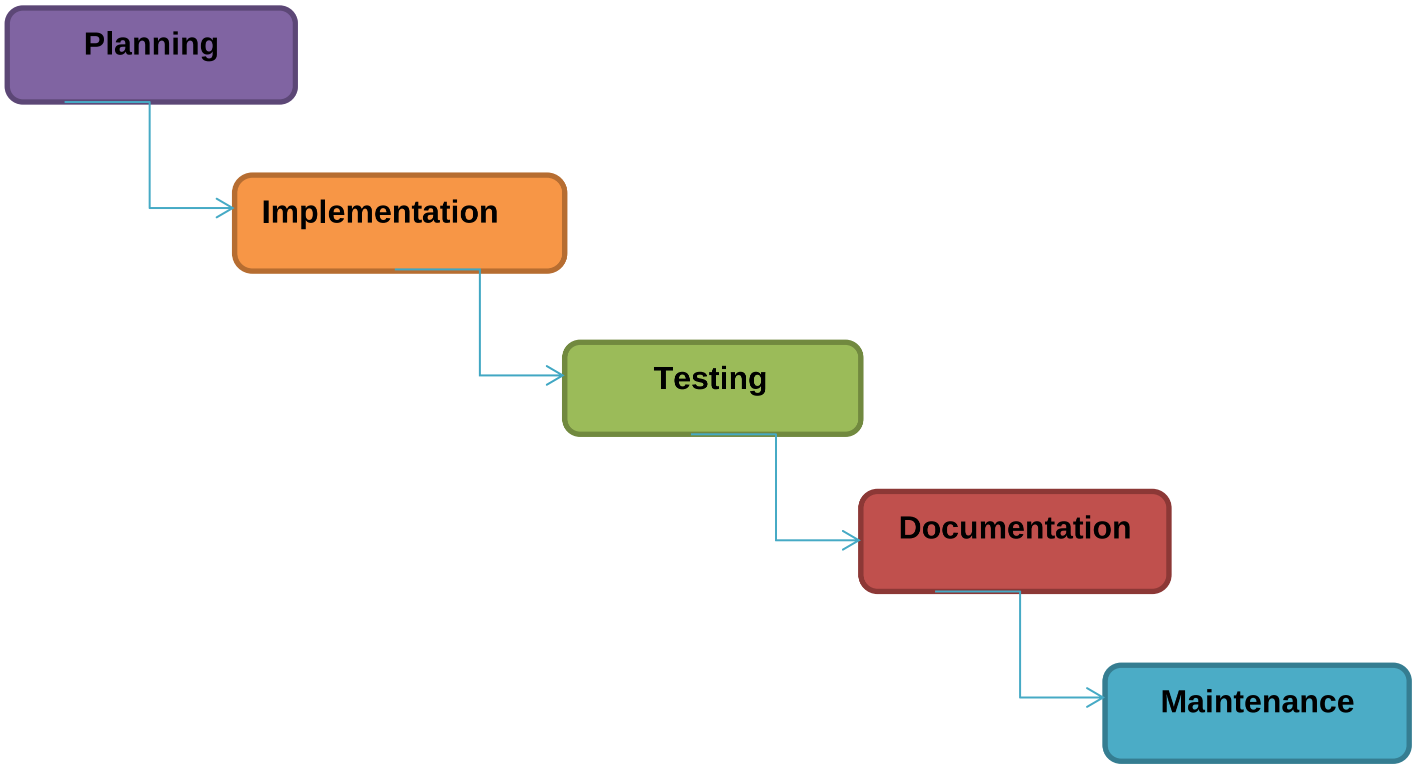
- ► DataBase:SQLLiet3
- ► django version:4.1
- ► python version:3.11.0
- ► OS:-Microsoft Windows
- ► Adobe Photoshop
- ► Plateform: VSCode

❖ HARDWARE REQUIREMENT

- ► Disk Space: 1GB
- ► RAM: 512MB
- ► Processor: 1.0GHz+
- ► CPU : 1.0 GHz

☯ SYSTEM DEVELOPMENT LIFE CYCLE (SDLC): ☯

The full form of SDLC is System Development Life Cycle. It is a framework that will defined during the software development the quality is maintain technically and all the requirement are fulfill accordingly.



- **Planning :-**

An experienced software engineers and primary level will collected information related to software make analysis about requirement like client of companies after connecting the information to make a documentation for the information scope in the software.

- **Implementation:-**

According to collection information once the targeting how to create aproject the coding code many limitations complexity and other problems will be solved and software will be completed.

- **Documentation :-**

When the project gets form one phase to another phase create a details reports or documentation for the future reference and API (Application Programming Interface) information.

☯ System Analysis ☯

Analysis is an important part of any project; is analysis is not done properly then whole project move in the wrong direction. It also provides a schedule for proper project work.

* Analysis task divided into 2 areas:

- ✓ Feasibility Study.
- ✓ Requirement Analysis.

➢ Feasibility Study:

Feasibility study of the system is a very important stage during website design. Feasibility study is a test of a website proposal according to its workability impact on the organization, ability to meet user needs, and effective use of resources. Feasibility study decides whether the website is properly developed or not.

Following aspects are taken into account during feasibility study.

- ✓ **<u>Technical Feasibility:</u>**
- ✓ **<u>Economic Feasibility:</u>**
- ✓ **<u>Operational Feasibility</u>:**

➢ —► Technical Feasibility:

-> It is not be affordable for the organization to employ new professionals then having a computer website thus the requirement makes it technical feasible.

-> The current set-up is sufficient for the processing of all kind of task.

➢ —► Economic Feasibility:

-> Implementation of this web-site will be a lifetime investment, which will ensure returns to the store of good services & market value through out the future. So the website is found economically feasible.

-> New website is user friendly.

-> Less time required for processing with manual website.

-> website will perform all kind of tasks very fast manual

➢ —► Operational Feasibility:

-> The web-site is easy to learn and it will require a very short time to learn the operation of the web-site for a person having knowledge about computer. So that website was operationally feasible.

-> As the web-site is made up in windows base environment so the graphical user interface made the web-site very user friendly and easy to operate.

-> The proposed system will fulfill the organization requirement.

➢ <u>Requirement Analysis:</u>

"These are the basic facilities that the good computer consumable project must require and all other information that I require with enhance of usage."

-> my website provide best speed.

-> Be Available Online Every Time.

-> Create Effective Strategy using Data Analysis.

☯ About Converse ☯

- **SITE URL: …………………………………**

- **This website is a live chat website this provide best communication platform for customer.**

- **This website is created using python with Django .**

- **To provide computerized data storage facility. We can search easily any record.**

- **The website is user friendly and anyone having computer knowledge can handle it easily.**

- **Suitabile for communicating and has a flawless easy login system with password recovery system.**

➢ Chatting Process :

1. Visit our website on your phone or computer.

2. Log in with valid credentials or create your new account.

3. Select the account you want to chat with.

4. If you want to chat with a group of people at a time you can select chatroom link.

5. Enter Your temporary username and the room name and enjoy chatting.

➢ Adantage of Converse:

- User can connect with people without going anywhere.

- It is convenient for users as this website is free of sponsored ads and any kind of service cost.

- The website is flexible to be used and for chatting with great UX.

☯ Dataflow Diagram ☯

About DFD (Data Flow Diagram):

The DFD (Data Flow diagram) is also known as Bubble chart. It is a simple graphical notation that can be used to represent a system in term of the input data to the system. Various processing carried out on this data, and the output data generated by the system. The main reason the DFD technique is so popular is on the account of the fact that it is very simple formulism-it is simple to understandd and use. A DFD model uses a very limited number of primitive symbols.

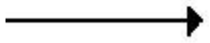
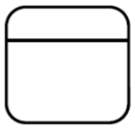
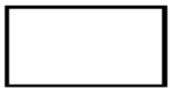
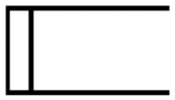
A data flow diagram is a graphical view of how data is processed in a system in terms of input and output.

The Data Flow Diagramming technique follows the very simple set of intuitive concepts and rules.

Data Flow Diagram (DFD) shows the flow of the data in to the system and processes and data stores. Data Flow Diagram has two types,

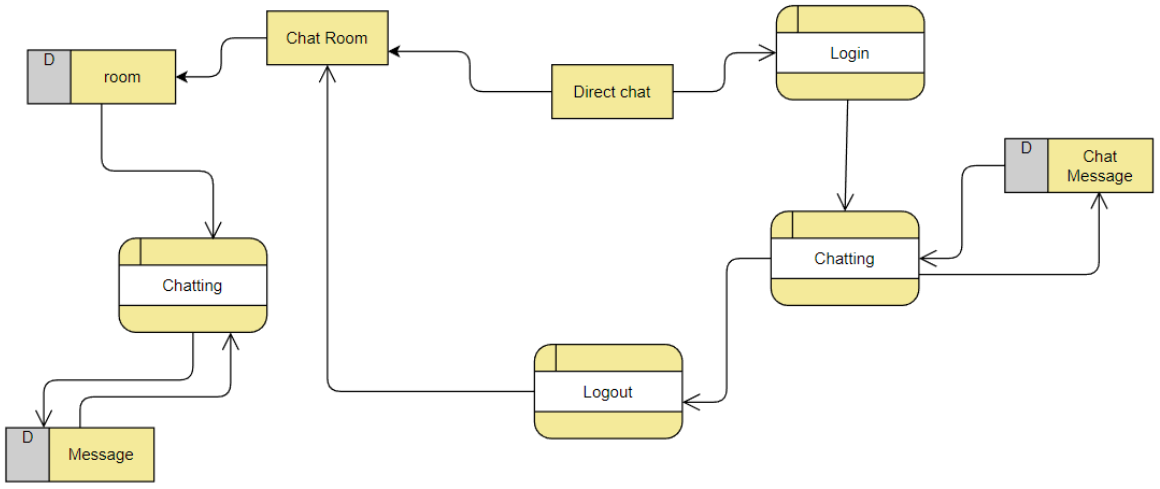
- Logical Data Flow Diagram
- Physical Data Flow Diagram

➢ Data flow diagram symbol:

Symbol	Description
	Data Flow – Data flow are pipelines through the packets of information flow.
	Process : A Process or task performed by the system.
	Entity : Entity are object of the system. A source or destination data of a system.
	Data Store : A place where data to be stored.

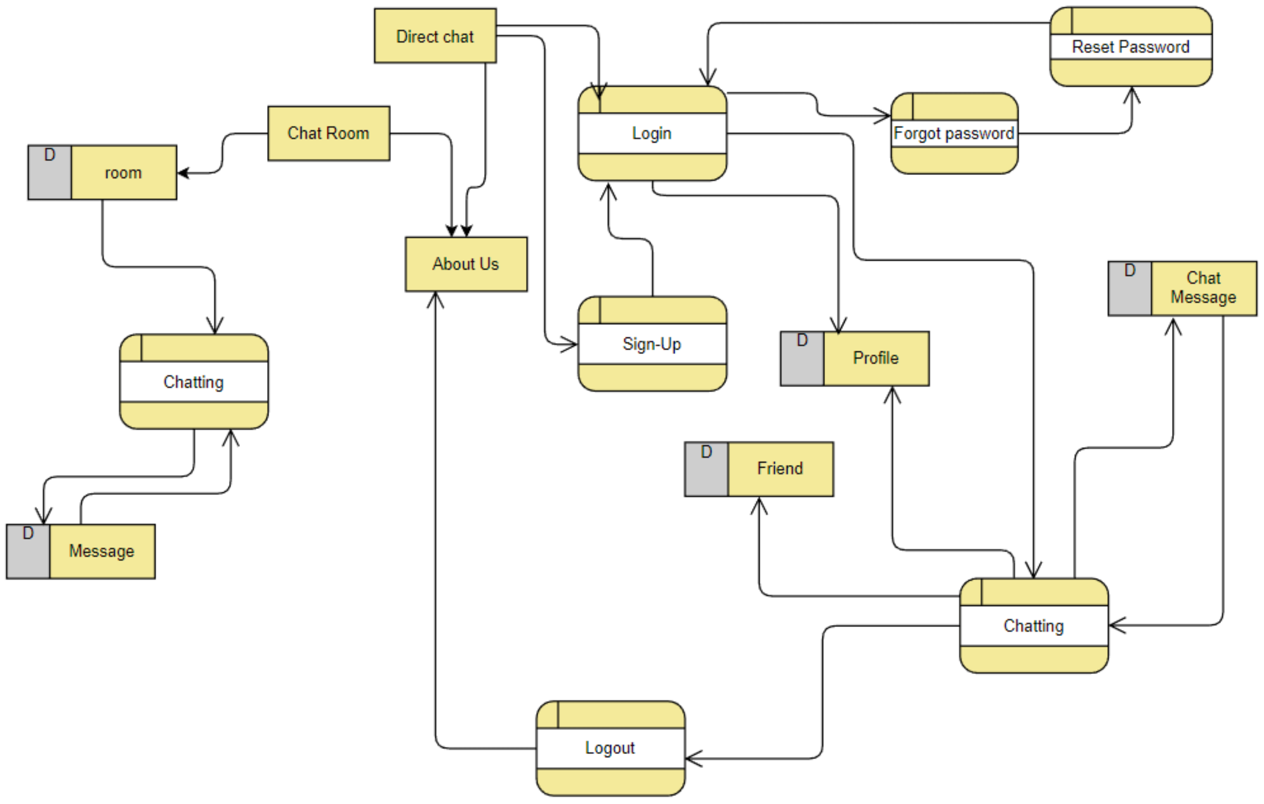
➢ <u>Zero Level DFD:</u>

DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities.



➢ <u>One Leval DFD:</u>

In 1-level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level DFD into subprocesses.



☯ Database (Table Layout) ☯

❖**Database :** sqlite3

❖ **Databse name:** db. Sqlite

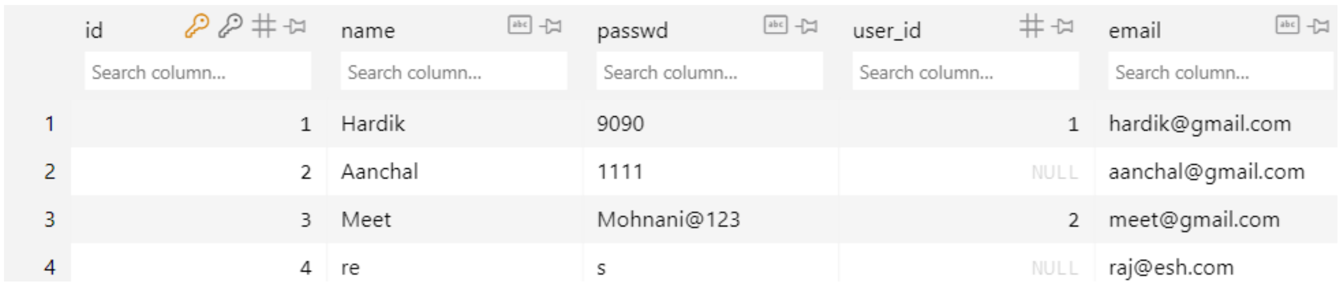
❖**MySQLite User Name:** hardik

❖**MySQLite User Password :** ********

sqlite3' in your project directory. The file is a database file that will keep all of the data that you will be generating. Since Django is a server-side framework, it treats your computer as the host when you run the server from the command line or terminal.

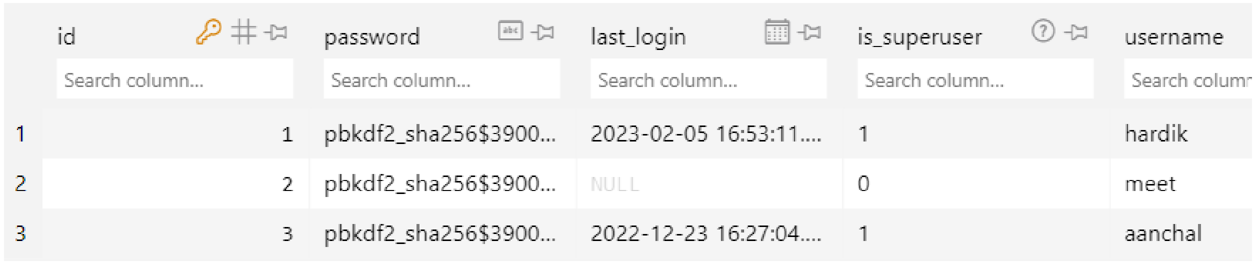
Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases. There is no need to install this module separately as it comes along with Python after the 4.1 version.

- **Live_chat_Profile**



	id	name	passwd	user_id	email
	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	Hardik	9090	1	hardik@gmail.com
2	2	Aanchal	1111	NULL	aanchal@gmail.com
3	3	Meet	Mohnani@123	2	meet@gmail.com
4	4	re	s	NULL	raj@esh.com

- **Auth_User:**



	id	password	last_login	is_superuser	username
	Search column...	Search column...	Search column...	Search column...	Search column
1	1	pbkdf2_sha256$3900...	2023-02-05 16:53:11....	1	hardik
2	2	pbkdf2_sha256$3900...	NULL	0	meet
3	3	pbkdf2_sha256$3900...	2022-12-23 16:27:04....	1	aanchal

- **Django_Admin_Log:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	20	Not null
Object_id	text	150	Not null

Object_repr	varchar	200	Null
Action_flag	smallint	255	Not null
Change_message	text	128	Not null
Content_type_id	int	150	Null
User_id	int	15	Not null
Action_time	datetime	254	Not null

- **Django_Migrations:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	20	Not null
app	varchar	255	Not null
name	varchar	255	Null

- **Django_Migrations:**

Field Name	Data Type	Size	Constraints
name	varchar	255	Not null
seq	int	255	Not null

- **Auth_Group_Permission:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	255	Not null
Group_id	int	255	Not null
Permission_id	int	255	Not null

- **Auth_User_Group:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	255	Not null
Group_id	int	255	Not null
Permission_id	int	255	Not null

- **Auth_User_User_Permission:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	255	Not null
Group_id	int	255	Not null
Permission_id	int	255	Not null

- **Django_Content_Type:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	20	Not null
App_label	varchar	100	Not null
Model	Varchar	100	Not null

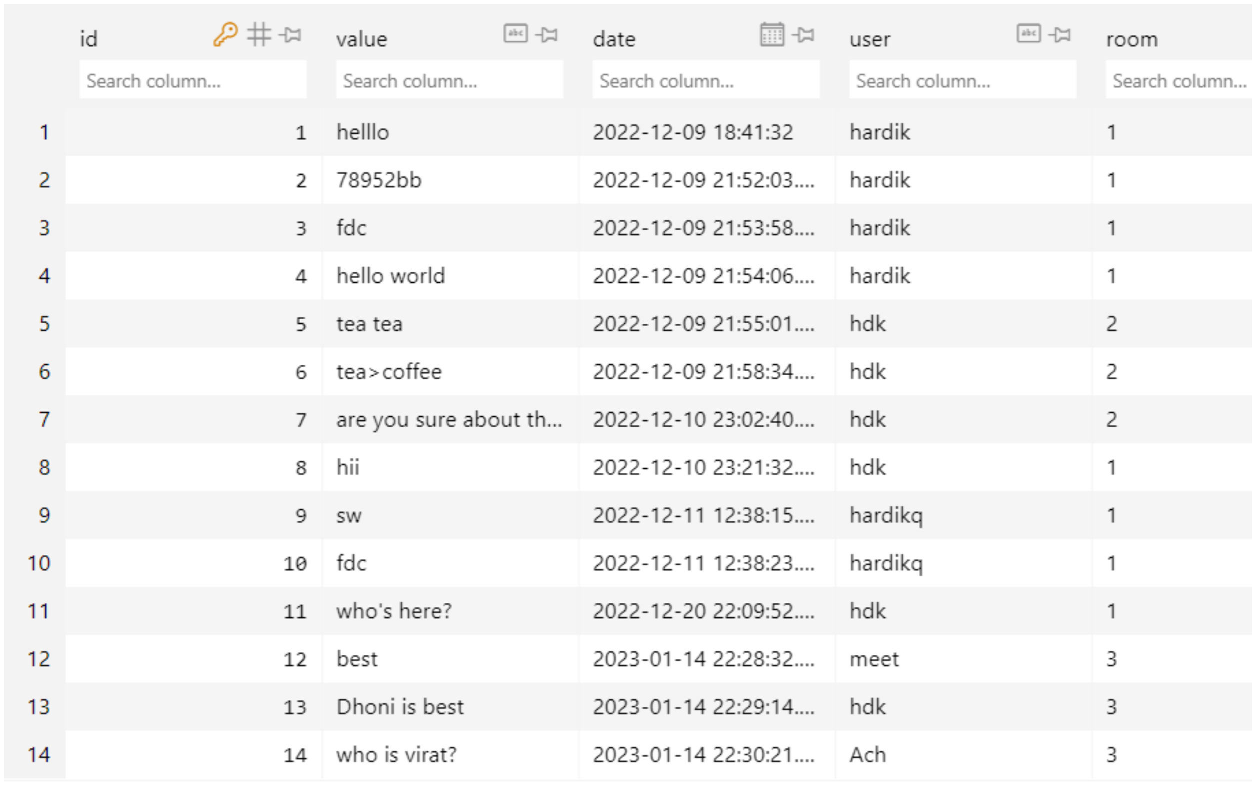
- **Auth_Group:**

Field Name	Data Type	Size	Constraints
ID(P.K)	int	20	Not null
name	varchar	150	Not null

- **Django_session:**

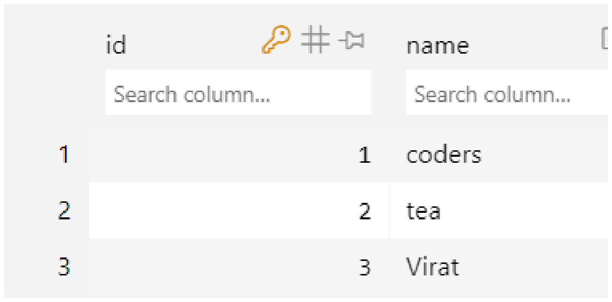
Field Name	Data Type	Size	Constraints
Session_Key(P.K)	varchar	40	Not null
Session_data	Text	100	Not null
Expire_date	datetime	100	Not null

- **chat_message**



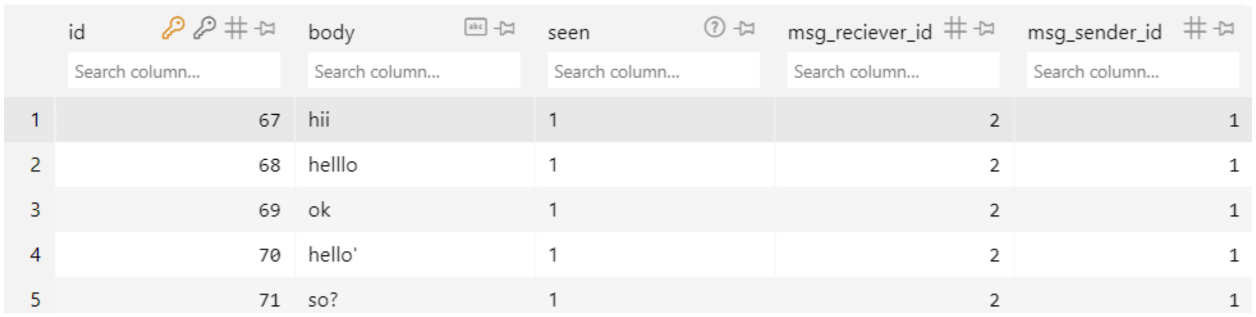
	id	value	date	user	room
	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	helllo	2022-12-09 18:41:32	hardik	1
2	2	78952bb	2022-12-09 21:52:03....	hardik	1
3	3	fdc	2022-12-09 21:53:58....	hardik	1
4	4	hello world	2022-12-09 21:54:06....	hardik	1
5	5	tea tea	2022-12-09 21:55:01....	hdk	2
6	6	tea>coffee	2022-12-09 21:58:34....	hdk	2
7	7	are you sure about th...	2022-12-10 23:02:40....	hdk	2
8	8	hii	2022-12-10 23:21:32....	hdk	1
9	9	sw	2022-12-11 12:38:15....	hardikq	1
10	10	fdc	2022-12-11 12:38:23....	hardikq	1
11	11	who's here?	2022-12-20 22:09:52....	hdk	1
12	12	best	2023-01-14 22:28:32....	meet	3
13	13	Dhoni is best	2023-01-14 22:29:14....	hdk	3
14	14	who is virat?	2023-01-14 22:30:21....	Ach	3

▪ chat_room



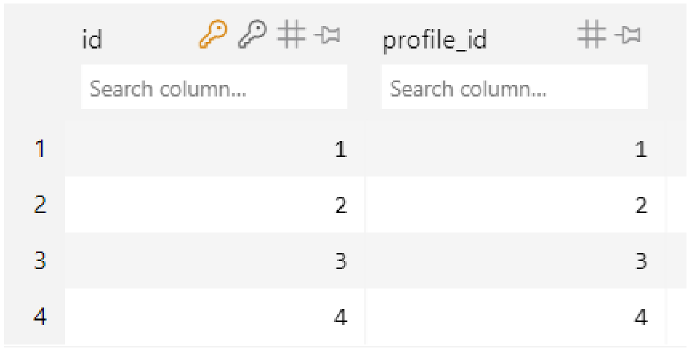
	id	name
	Search column...	Search column...
1	1	coders
2	2	tea
3	3	Virat

▪ live_chat_messages



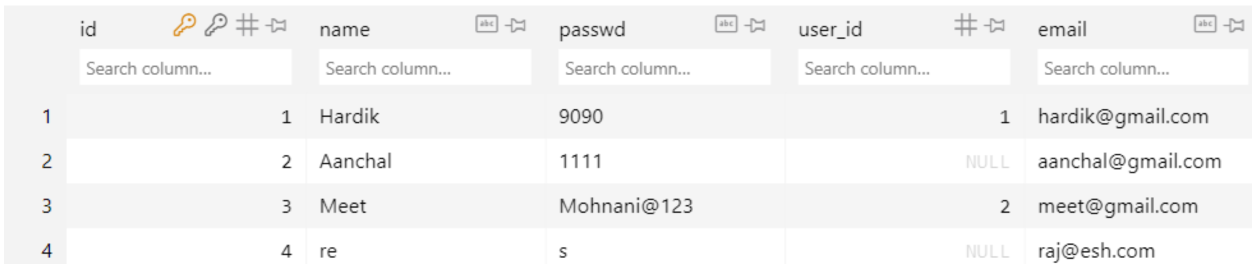
	id	body	seen	msg_reciever_id	msg_sender_id
	Search column...	Search column...	Search column...	Search column...	Search column...
1	67	hii	1	2	1
2	68	helllo	1	2	1
3	69	ok	1	2	1
4	70	hello'	1	2	1
5	71	so?	1	2	1

▪ live_chat_friends



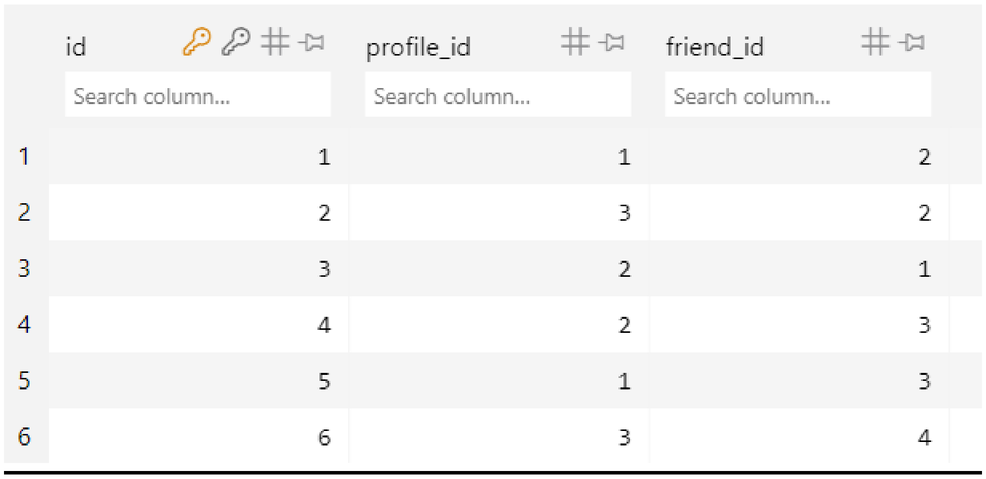
	id	profile_id
	Search column...	Search column...
1	1	1
2	2	2
3	3	3
4	4	4

▪ live_chat_profile



	id	name	passwd	user_id	email
	Search column...	Search column...	Search column...	Search column...	Search column...
1	1	Hardik	9090	1	hardik@gmail.com
2	2	Aanchal	1111	NULL	aanchal@gmail.com
3	3	Meet	Mohnani@123	2	meet@gmail.com
4	4	re	s	NULL	raj@esh.com

▪ live_chat_profile_friends



	id	profile_id	friend_id
	Search column...	Search column...	Search column...
1	1	1	2
2	2	3	2
3	3	2	1
4	4	2	3
5	5	1	3
6	6	3	4

☯ File Manager : ☯

Converse :

- Converse
  - ✓ __init__.py
  - ✓ asgi.py
  - ✓ settings.py
  - ✓ urls.py
  - ✓ wsgi.py

- Chat
  - __init__.py
  - A dmin.py
  - apps.py
  - models.py
  - tests.py
  - urls.py
  - views.py

- Live_chat
  - __init__.py
  - admin.py

        - apps.py
        - models.py
        - tests.py
        - urls.py
        - views.py
- Templates
    - chat
        - base.html
        - index.html
        - room.html
    - live_chat
        - 1.jpg
        - 2.jpg
        - 3.jpg
        - about.html
        - detail.html
        - forgot.html
        - index.html
        - login.html
        - reset.html
- Db.sqlite3
- Manage.py

☯ CMD Command's ☯

1. Python manage.py makemigrations:

makemigrations is responsible for packaging up your model changes into individual migration files - analogous to commits - and migrate is responsible for applying those to your database.

2. python manage.py migrate:

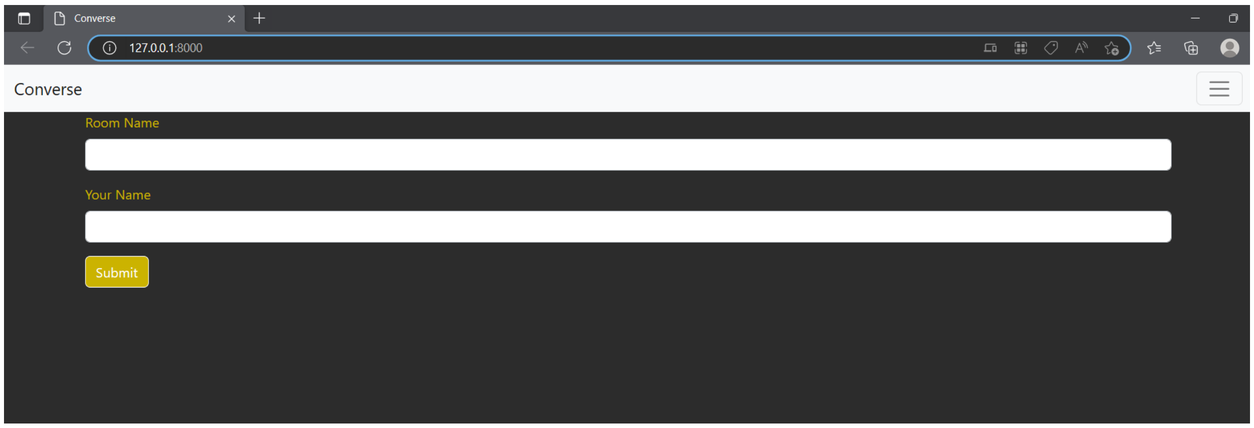
Migrations are Django's way of propagating changes you make to your models (adding a field, deleting a model, etc.) into your database schema. They're designed to be mostly automatic, but you'll need to know when to make migrations, when to run them, and the common problems you might run into.

3.python manage.py runserver:

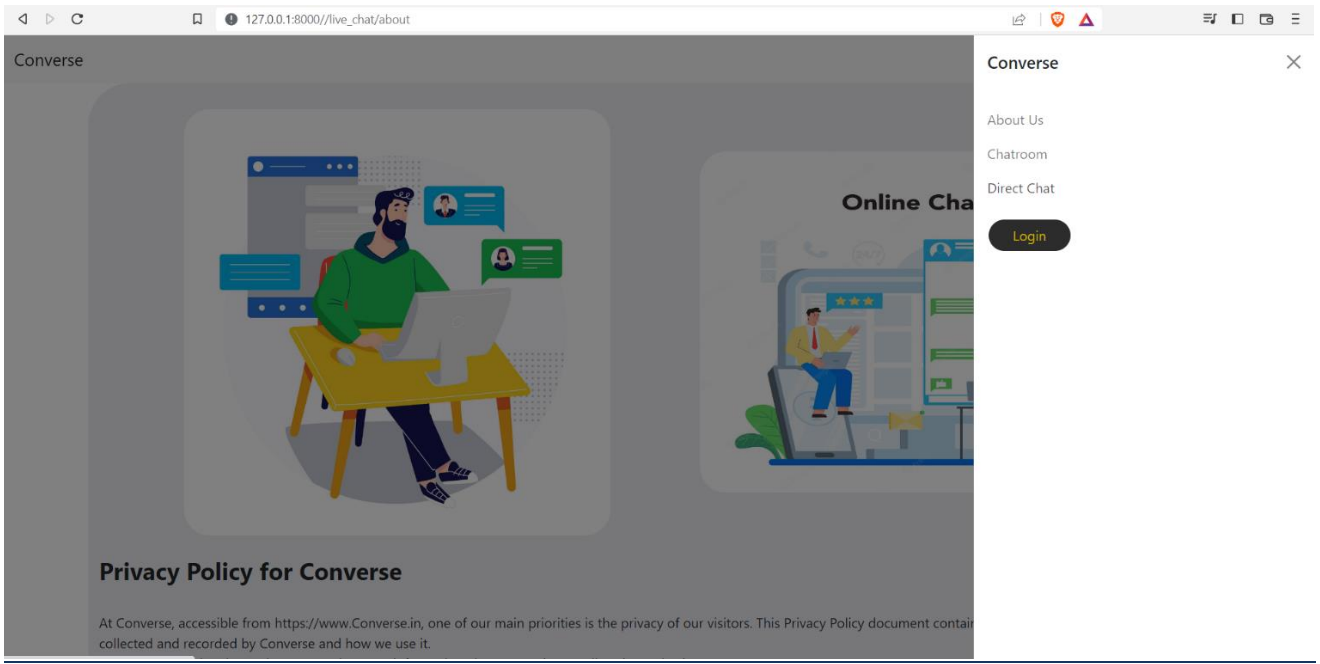
The runserver command is a built-in subcommand of Django's manage.py file that will start up a development server for this specific Django project.

☯ Website Screenshots ☯

CHAT ROOM



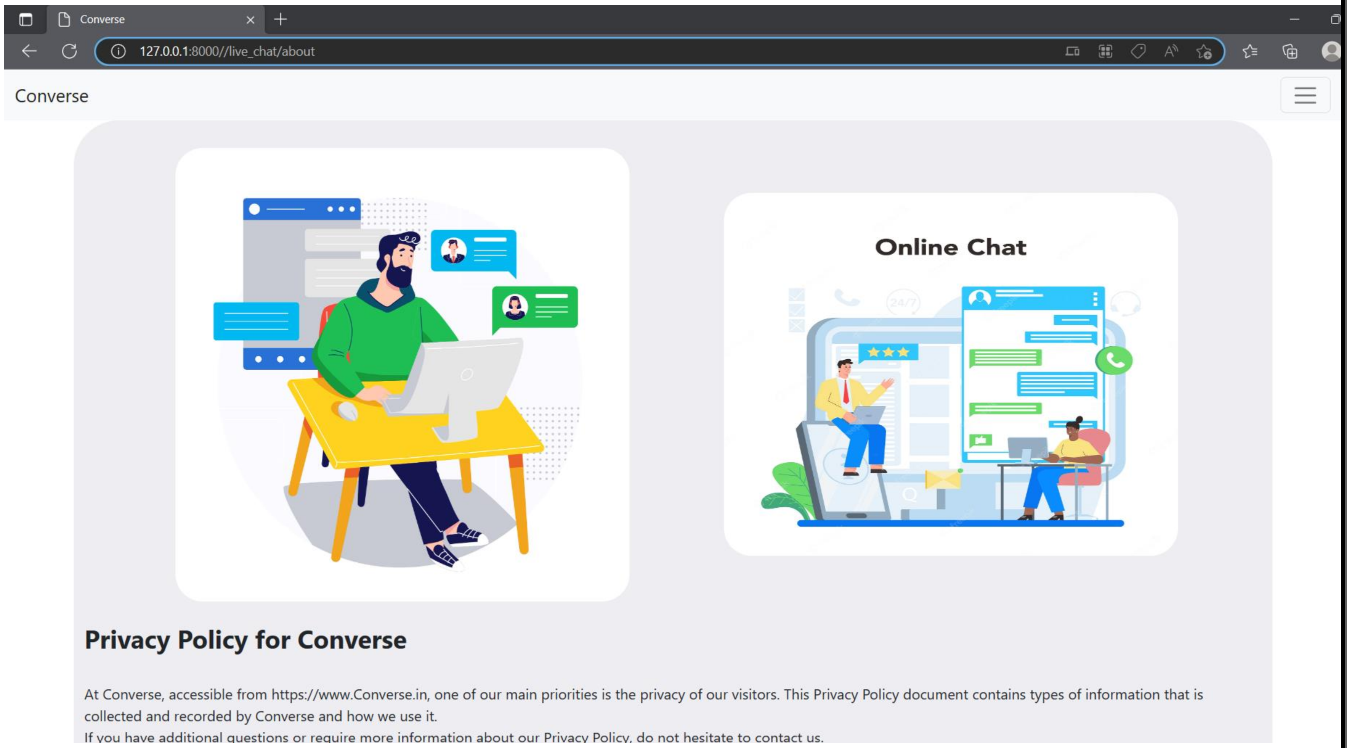
NAVIGATION BAR



Privacy Policy for Converse

At Converse, accessible from https://www.Converse.in, one of our main priorities is the privacy of our visitors. This Privacy Policy document contai
collected and recorded by Converse and how we use it.

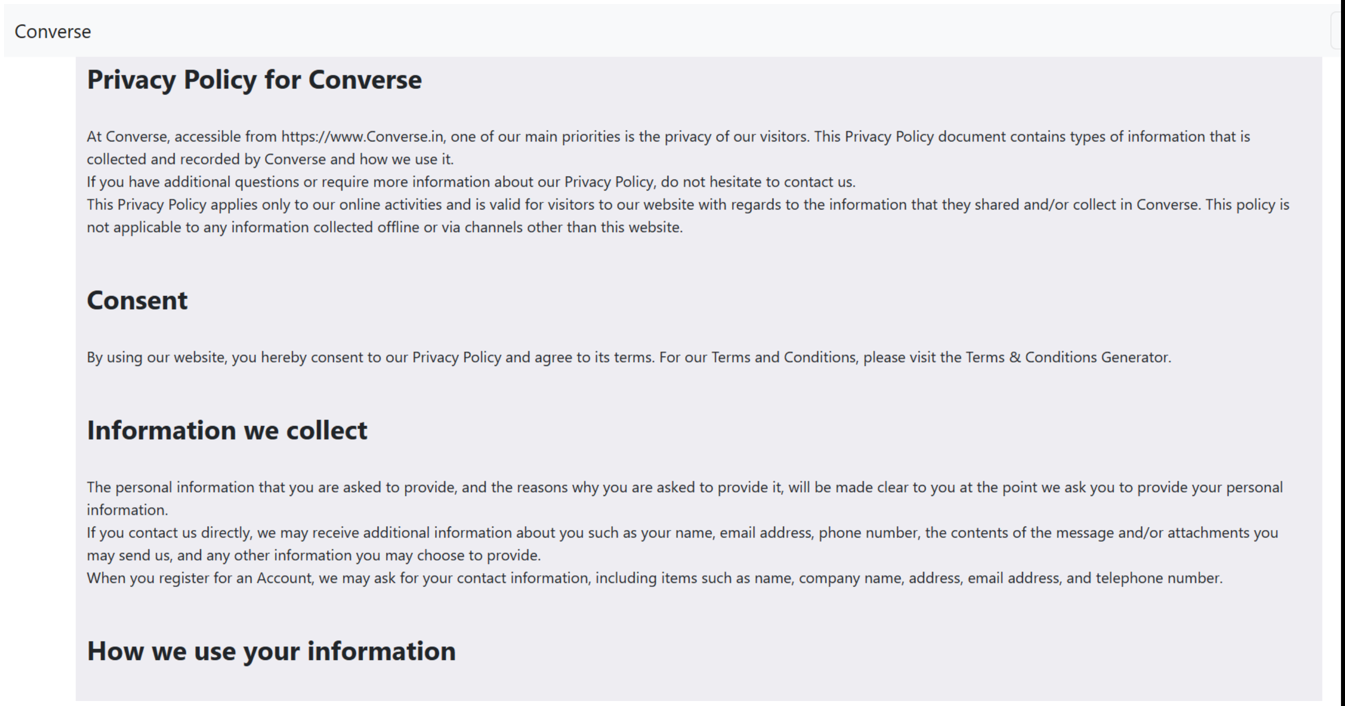
ABOUT US / PRIVACY POLICY



Privacy Policy for Converse

At Converse, accessible from https://www.Converse.in, one of our main priorities is the privacy of our visitors. This Privacy Policy document contains types of information that is collected and recorded by Converse and how we use it.

If you have additional questions or require more information about our Privacy Policy, do not hesitate to contact us.



Converse

Privacy Policy for Converse

At Converse, accessible from https://www.Converse.in, one of our main priorities is the privacy of our visitors. This Privacy Policy document contains types of information that is collected and recorded by Converse and how we use it.
If you have additional questions or require more information about our Privacy Policy, do not hesitate to contact us.
This Privacy Policy applies only to our online activities and is valid for visitors to our website with regards to the information that they shared and/or collect in Converse. This policy is not applicable to any information collected offline or via channels other than this website.

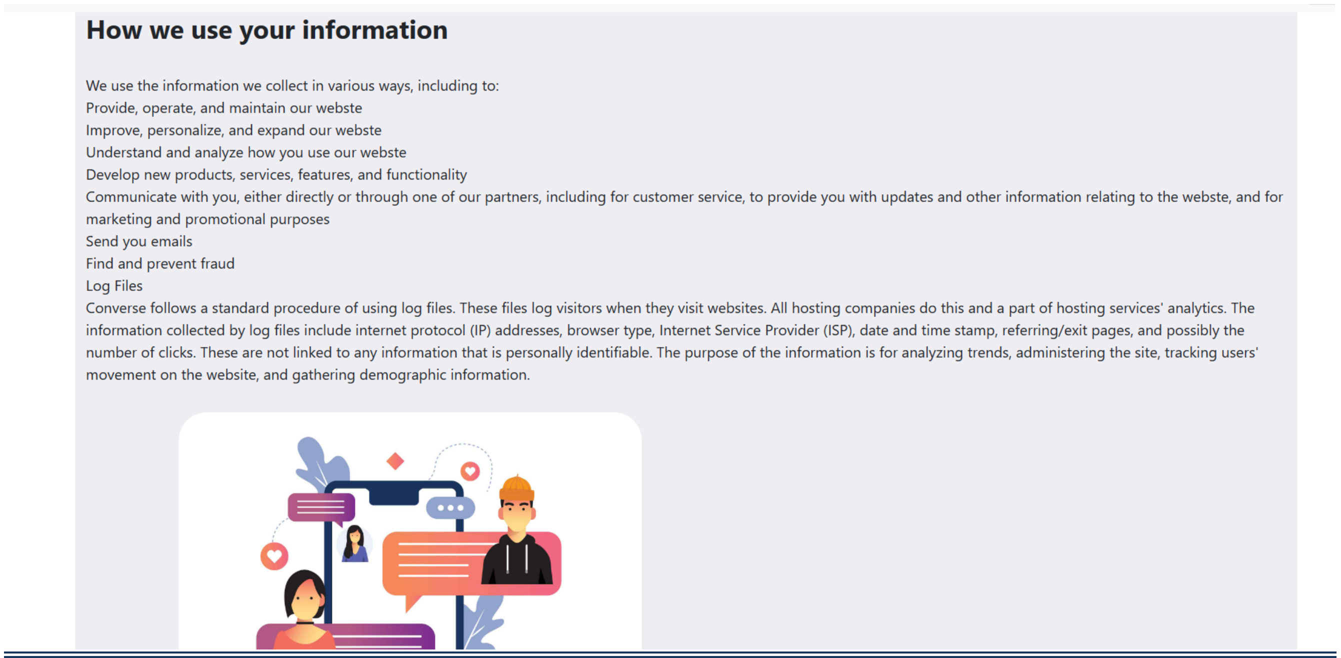
Consent

By using our website, you hereby consent to our Privacy Policy and agree to its terms. For our Terms and Conditions, please visit the Terms & Conditions Generator.

Information we collect

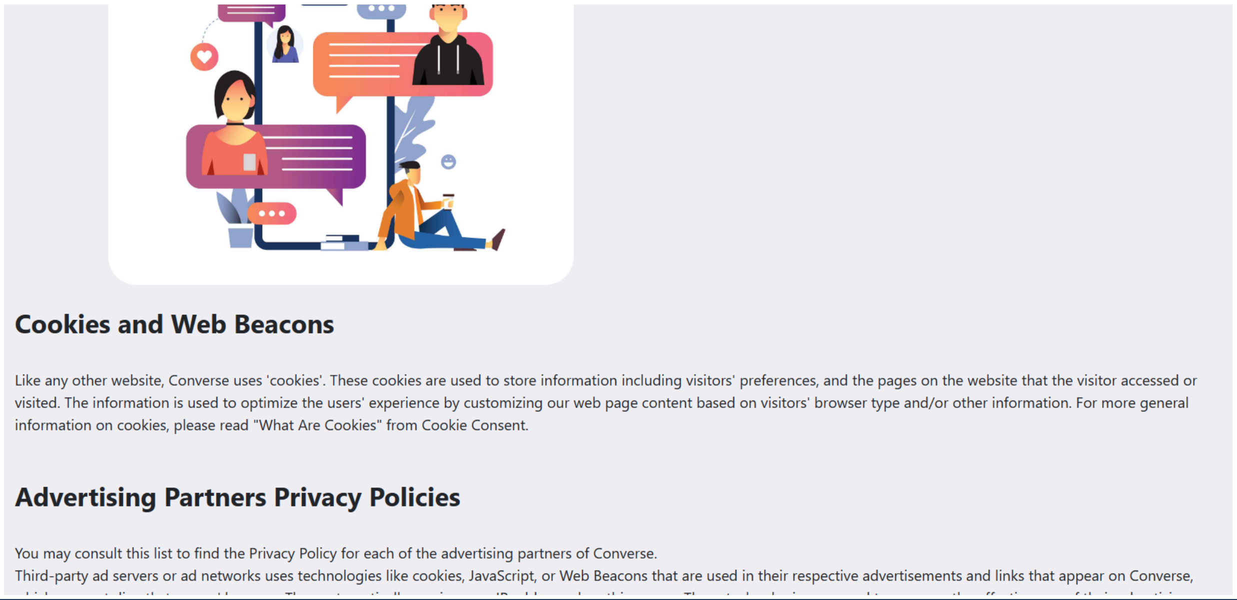
The personal information that you are asked to provide, and the reasons why you are asked to provide it, will be made clear to you at the point we ask you to provide your personal information.
If you contact us directly, we may receive additional information about you such as your name, email address, phone number, the contents of the message and/or attachments you may send us, and any other information you may choose to provide.
When you register for an Account, we may ask for your contact information, including items such as name, company name, address, email address, and telephone number.

How we use your information



How we use your information

We use the information we collect in various ways, including to:
Provide, operate, and maintain our webste
Improve, personalize, and expand our webste
Understand and analyze how you use our webste
Develop new products, services, features, and functionality
Communicate with you, either directly or through one of our partners, including for customer service, to provide you with updates and other information relating to the webste, and for marketing and promotional purposes
Send you emails
Find and prevent fraud
Log Files
Converse follows a standard procedure of using log files. These files log visitors when they visit websites. All hosting companies do this and a part of hosting services' analytics. The information collected by log files include internet protocol (IP) addresses, browser type, Internet Service Provider (ISP), date and time stamp, referring/exit pages, and possibly the number of clicks. These are not linked to any information that is personally identifiable. The purpose of the information is for analyzing trends, administering the site, tracking users' movement on the website, and gathering demographic information.

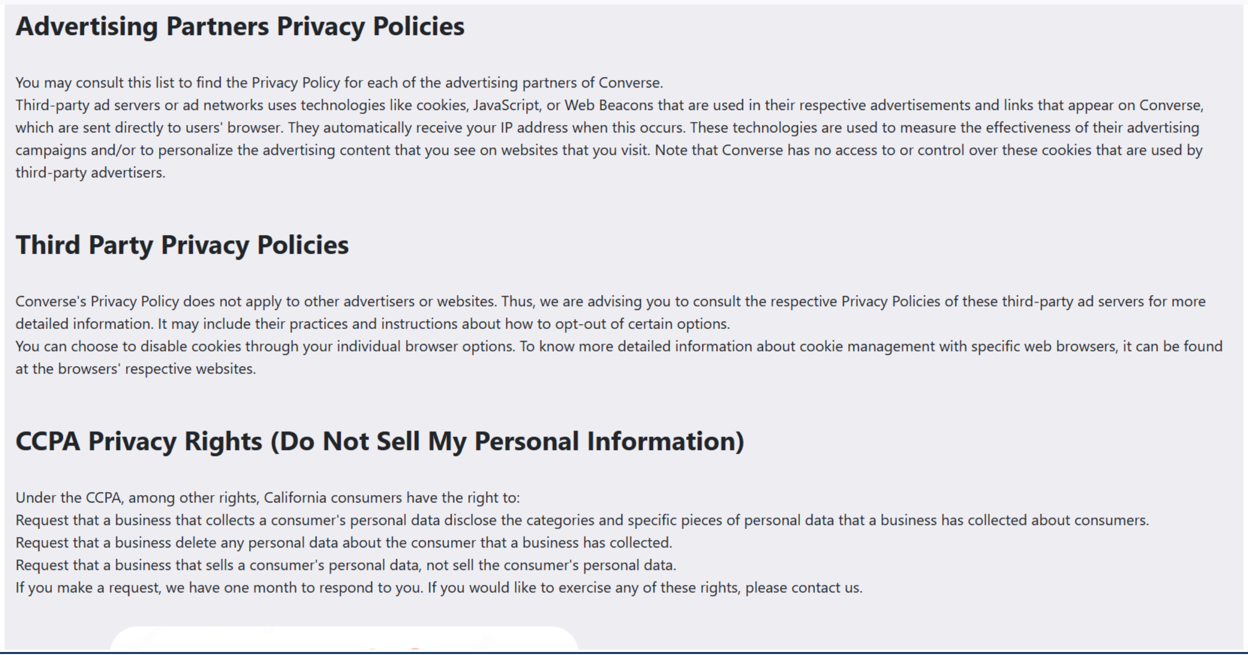


Cookies and Web Beacons

Like any other website, Converse uses 'cookies'. These cookies are used to store information including visitors' preferences, and the pages on the website that the visitor accessed or visited. The information is used to optimize the users' experience by customizing our web page content based on visitors' browser type and/or other information. For more general information on cookies, please read "What Are Cookies" from Cookie Consent.

Advertising Partners Privacy Policies

You may consult this list to find the Privacy Policy for each of the advertising partners of Converse.
Third-party ad servers or ad networks uses technologies like cookies, JavaScript, or Web Beacons that are used in their respective advertisements and links that appear on Converse,



Advertising Partners Privacy Policies

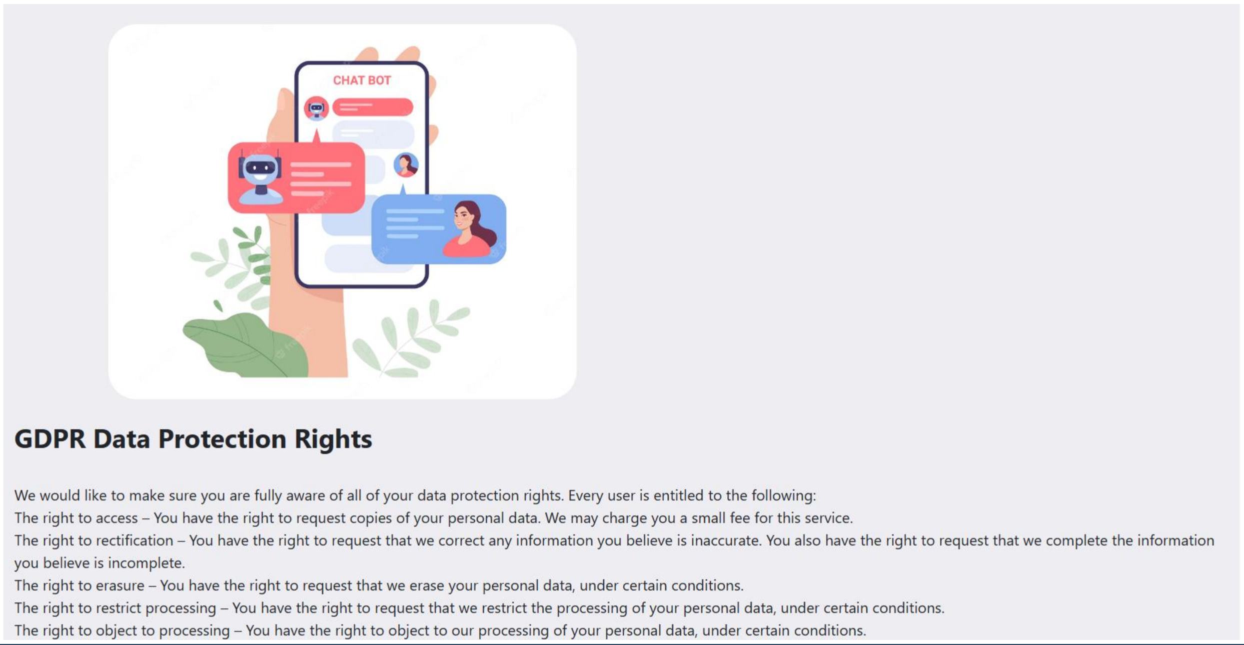
You may consult this list to find the Privacy Policy for each of the advertising partners of Converse.
Third-party ad servers or ad networks uses technologies like cookies, JavaScript, or Web Beacons that are used in their respective advertisements and links that appear on Converse, which are sent directly to users' browser. They automatically receive your IP address when this occurs. These technologies are used to measure the effectiveness of their advertising campaigns and/or to personalize the advertising content that you see on websites that you visit. Note that Converse has no access to or control over these cookies that are used by third-party advertisers.

Third Party Privacy Policies

Converse's Privacy Policy does not apply to other advertisers or websites. Thus, we are advising you to consult the respective Privacy Policies of these third-party ad servers for more detailed information. It may include their practices and instructions about how to opt-out of certain options.
You can choose to disable cookies through your individual browser options. To know more detailed information about cookie management with specific web browsers, it can be found at the browsers' respective websites.

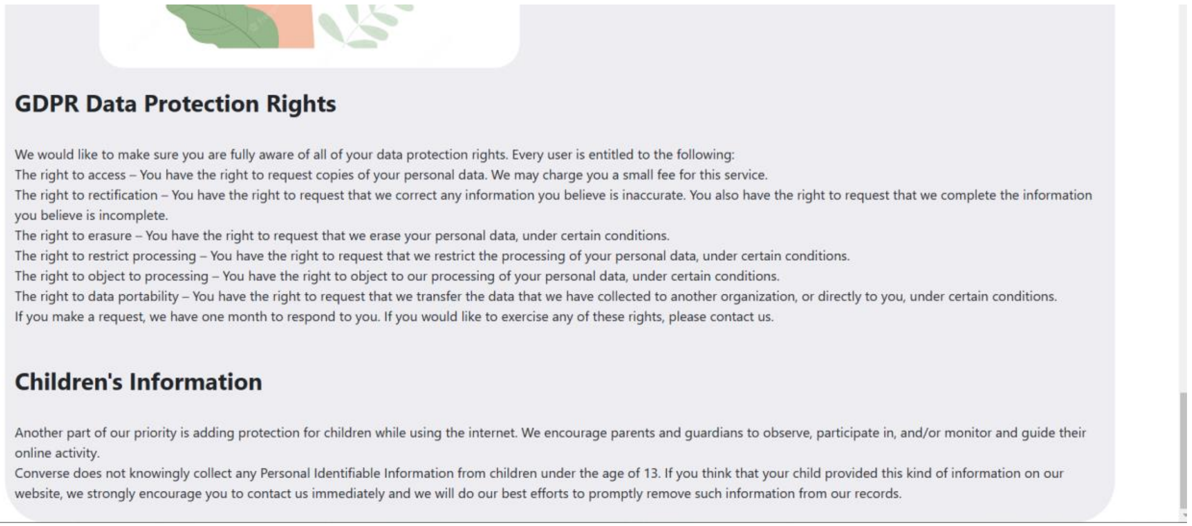
CCPA Privacy Rights (Do Not Sell My Personal Information)

Under the CCPA, among other rights, California consumers have the right to:
Request that a business that collects a consumer's personal data disclose the categories and specific pieces of personal data that a business has collected about consumers.
Request that a business delete any personal data about the consumer that a business has collected.
Request that a business that sells a consumer's personal data, not sell the consumer's personal data.
If you make a request, we have one month to respond to you. If you would like to exercise any of these rights, please contact us.



GDPR Data Protection Rights

We would like to make sure you are fully aware of all of your data protection rights. Every user is entitled to the following:
The right to access – You have the right to request copies of your personal data. We may charge you a small fee for this service.
The right to rectification – You have the right to request that we correct any information you believe is inaccurate. You also have the right to request that we complete the information you believe is incomplete.
The right to erasure – You have the right to request that we erase your personal data, under certain conditions.
The right to restrict processing – You have the right to request that we restrict the processing of your personal data, under certain conditions.
The right to object to processing – You have the right to object to our processing of your personal data, under certain conditions.



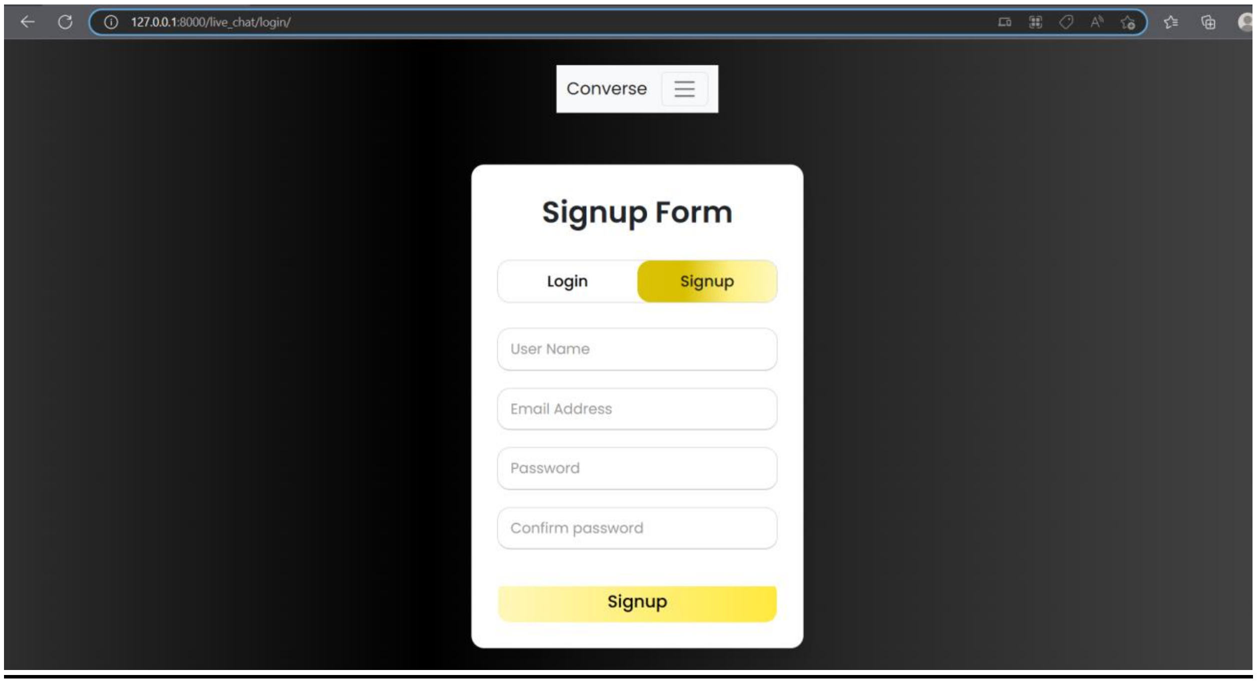
GDPR Data Protection Rights

We would like to make sure you are fully aware of all of your data protection rights. Every user is entitled to the following:
The right to access – You have the right to request copies of your personal data. We may charge you a small fee for this service.
The right to rectification – You have the right to request that we correct any information you believe is inaccurate. You also have the right to request that we complete the information you believe is incomplete.
The right to erasure – You have the right to request that we erase your personal data, under certain conditions.
The right to restrict processing – You have the right to request that we restrict the processing of your personal data, under certain conditions.
The right to object to processing – You have the right to object to our processing of your personal data, under certain conditions.
The right to data portability – You have the right to request that we transfer the data that we have collected to another organization, or directly to you, under certain conditions.
If you make a request, we have one month to respond to you. If you would like to exercise any of these rights, please contact us.

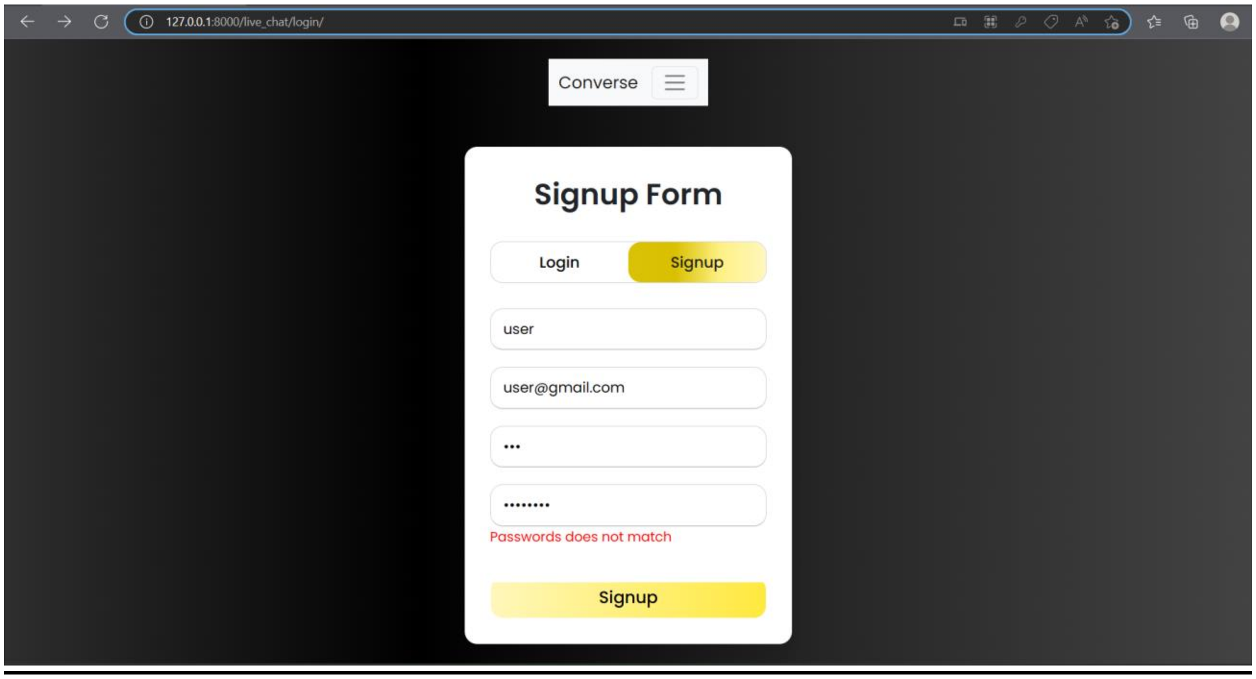
Children's Information

Another part of our priority is adding protection for children while using the internet. We encourage parents and guardians to observe, participate in, and/or monitor and guide their online activity.
Converse does not knowingly collect any Personal Identifiable Information from children under the age of 13. If you think that your child provided this kind of information on our website, we strongly encourage you to contact us immediately and we will do our best efforts to promptly remove such information from our records.

SIGN-UP


127.0.0.1:8000/live_chat/login/

Signup Form


127.0.0.1:8000/live_chat/login/

Signup Form

user

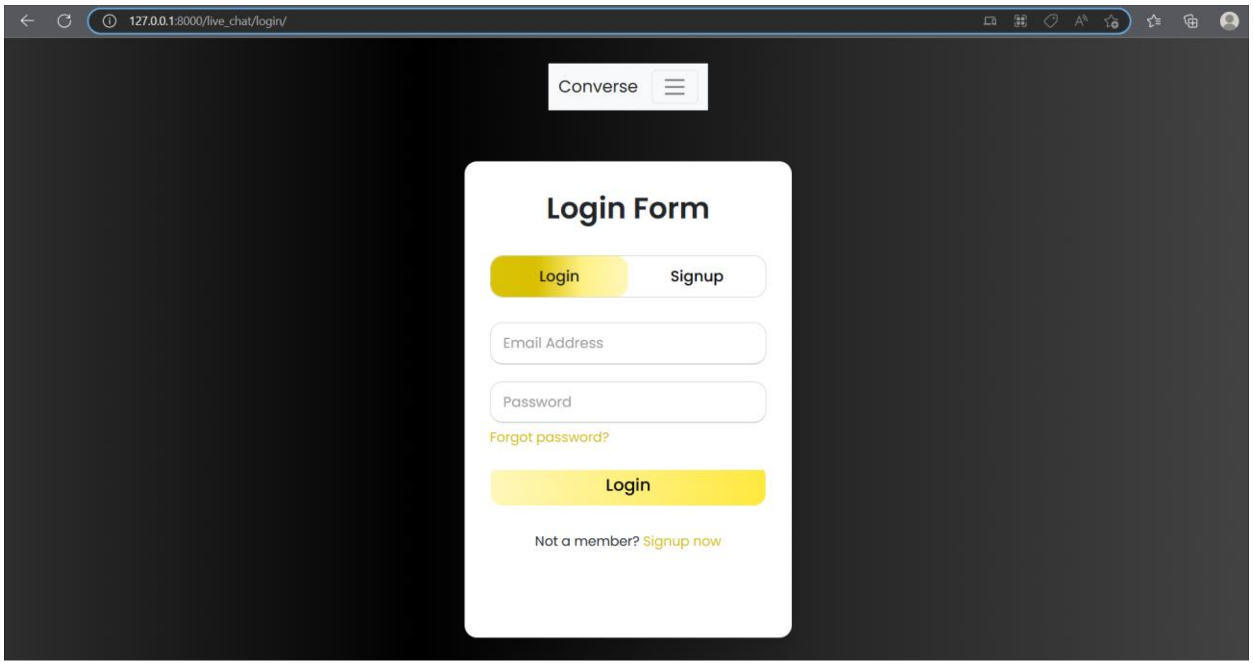
user@gmail.com

•••

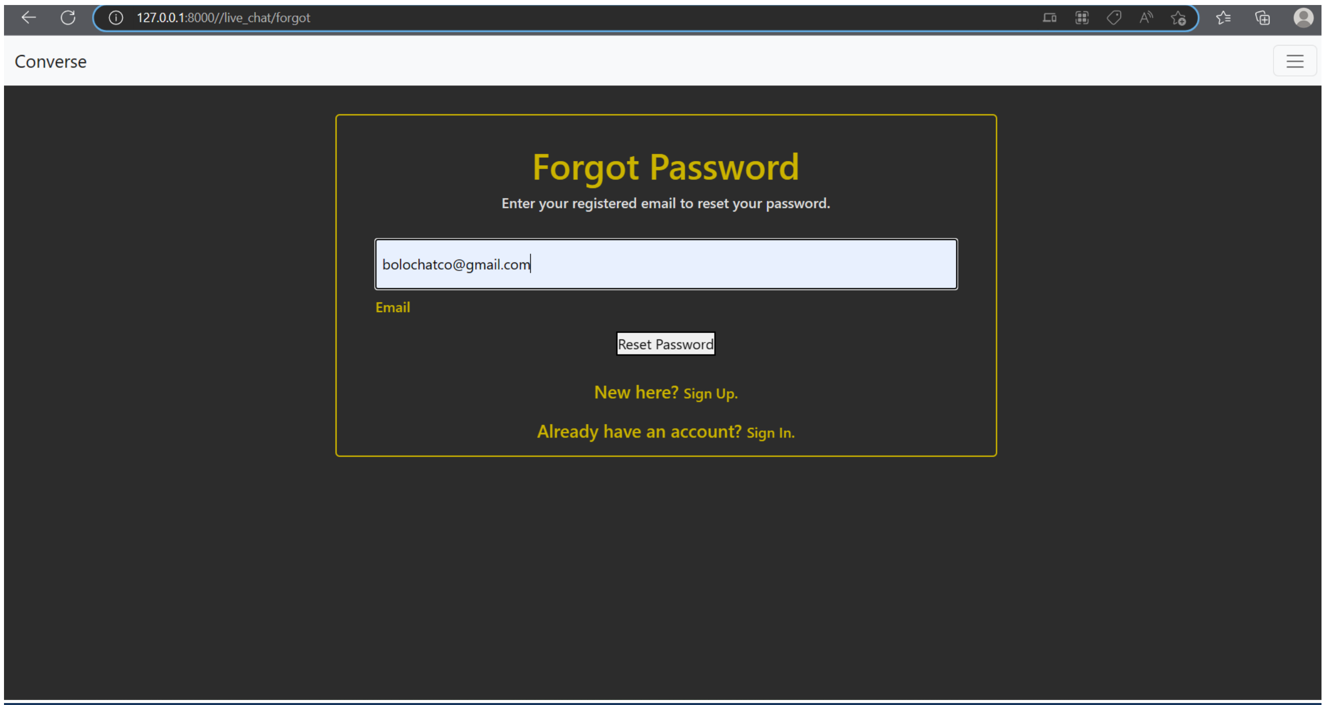
••••••••

Passwords does not match

LOGIN



FORGOT PASSWORD



127.0.0.1:8000//live_chat/forgot

Converse

Forgot Password

Enter your registered email to reset your password.

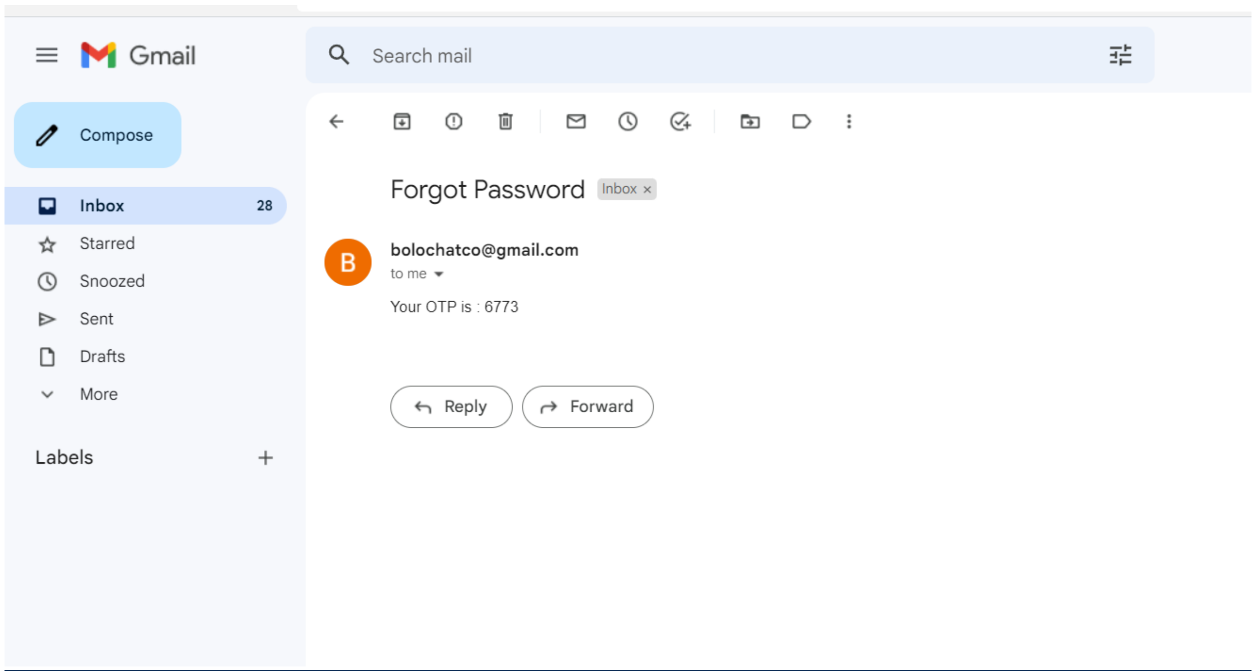
bolochatco@gmail.com

Email

Reset Password

New here? Sign Up.

Already have an account? Sign In.

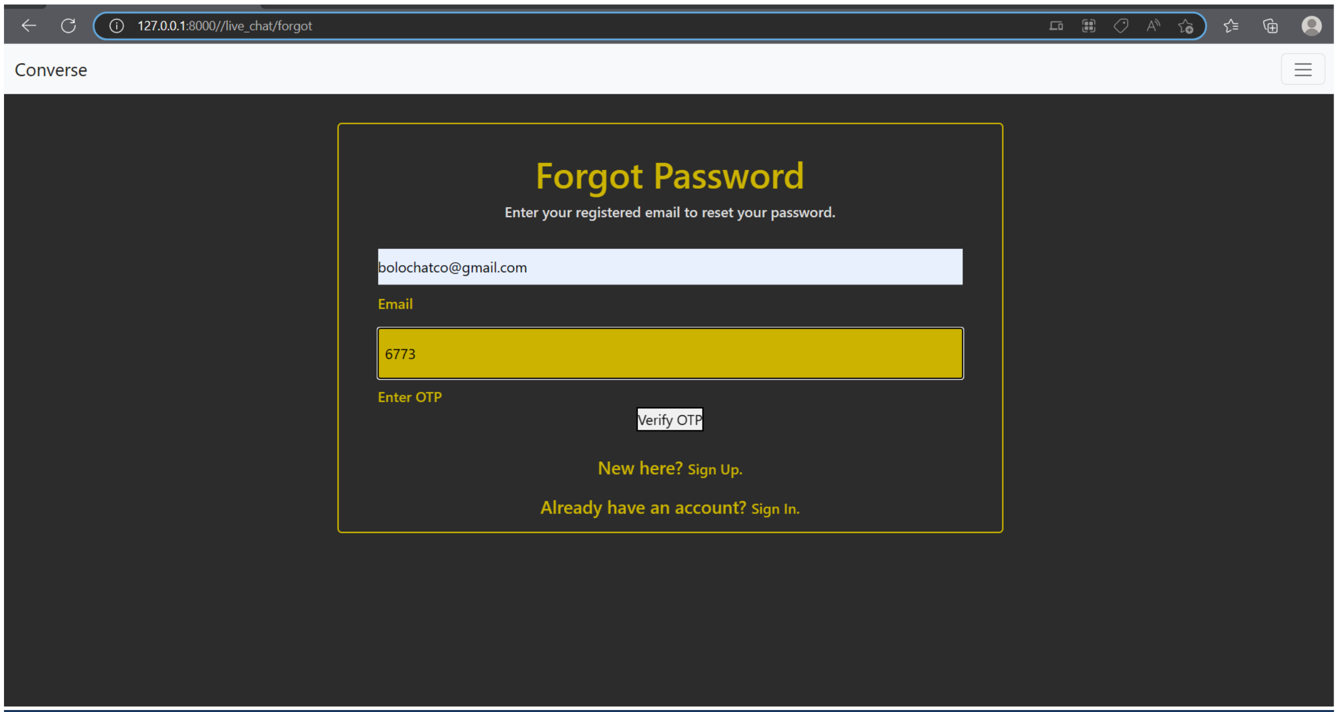


Forgot Password Inbox

bolochatco@gmail.com

to me

Your OTP is : 6773



127.0.0.1:8000//live_chat/forgot

Converse

Forgot Password

Enter your registered email to reset your password.

bolochatco@gmail.com

Email

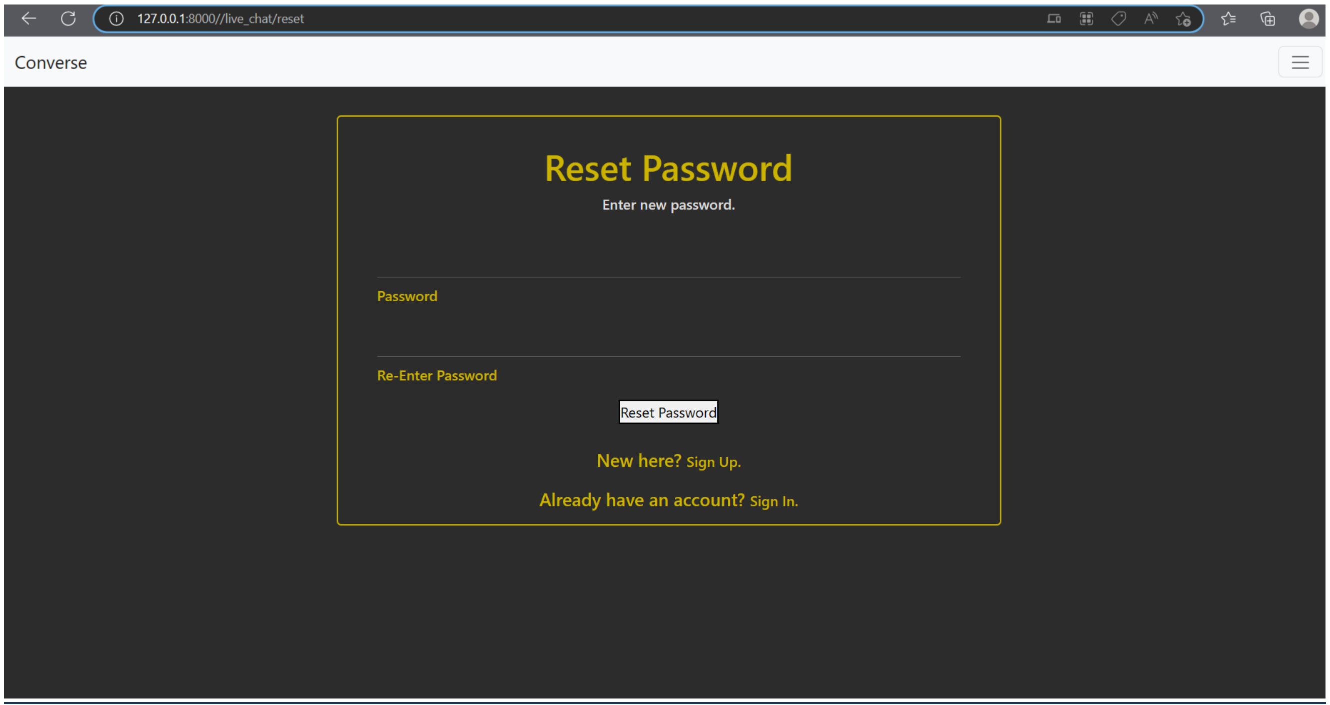
6773

Enter OTP

Verify OTP

New here? Sign Up.

Already have an account? Sign In.



127.0.0.1:8000//live_chat/reset

Converse

Reset Password

Enter new password.

Password

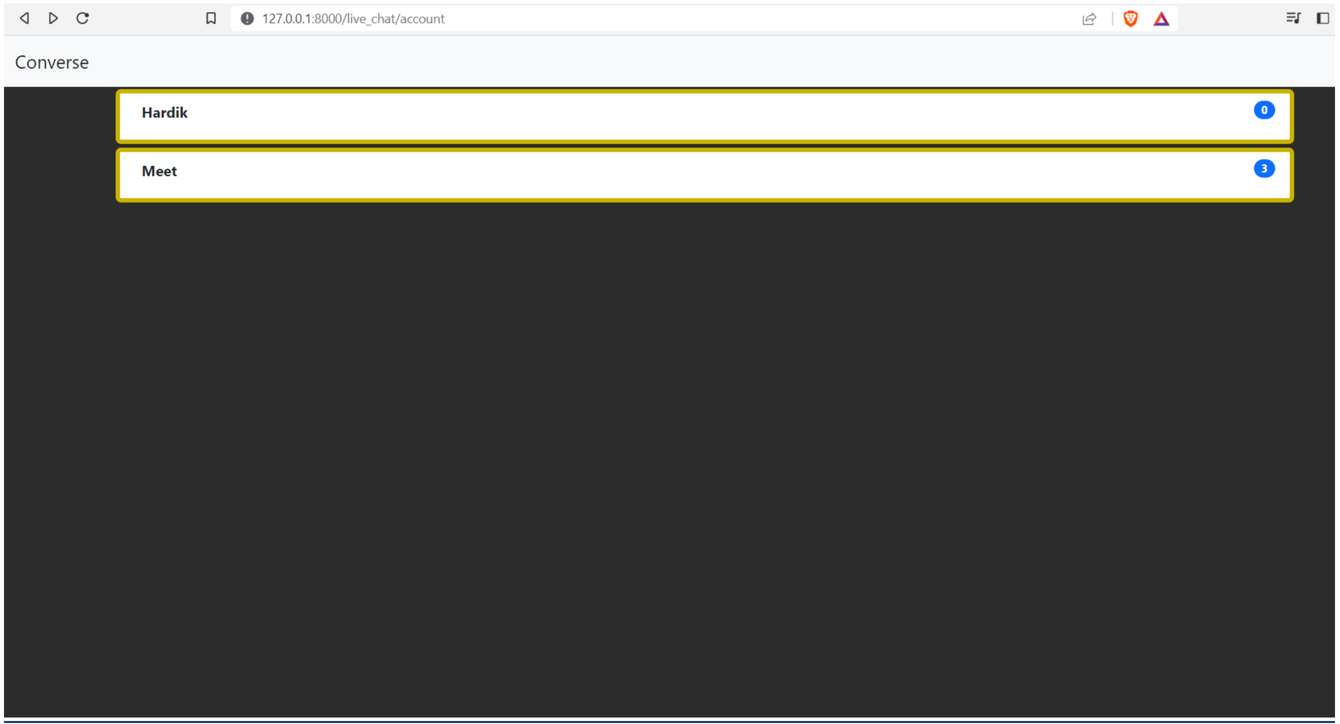
Re-Enter Password

Reset Password

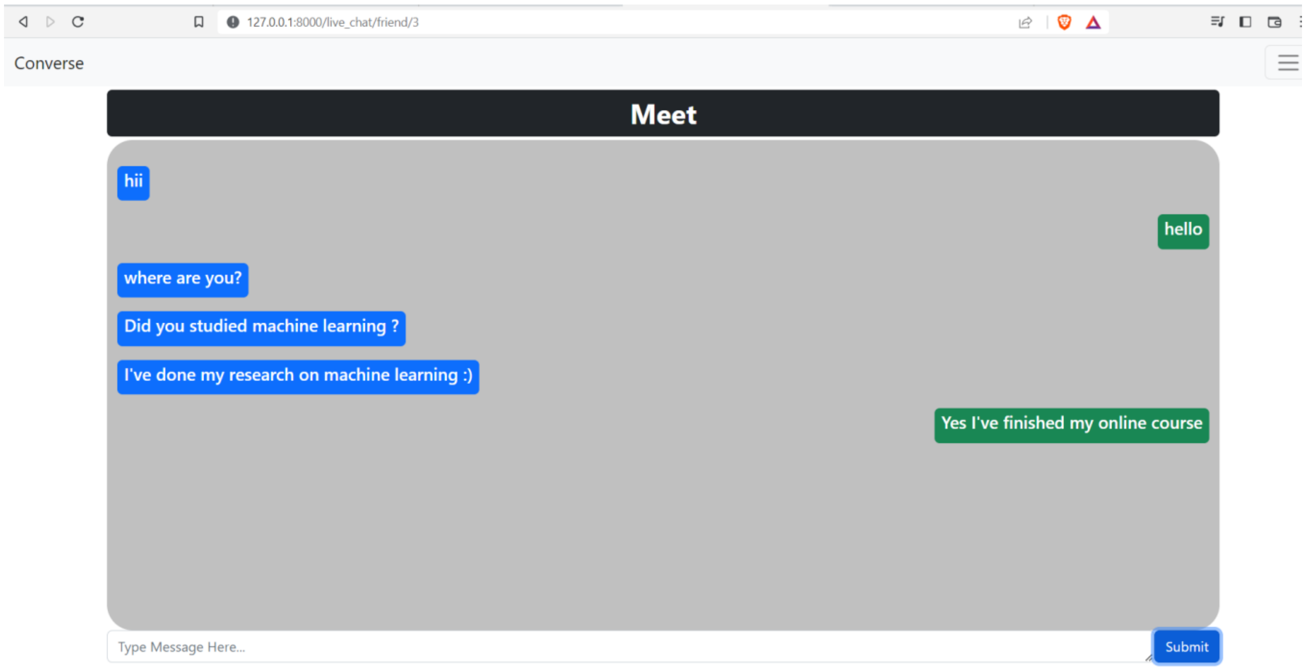
New here? Sign Up.

Already have an account? Sign In.

ACCOUNT (with notifications)



DIRECT CHAT



hii

hello

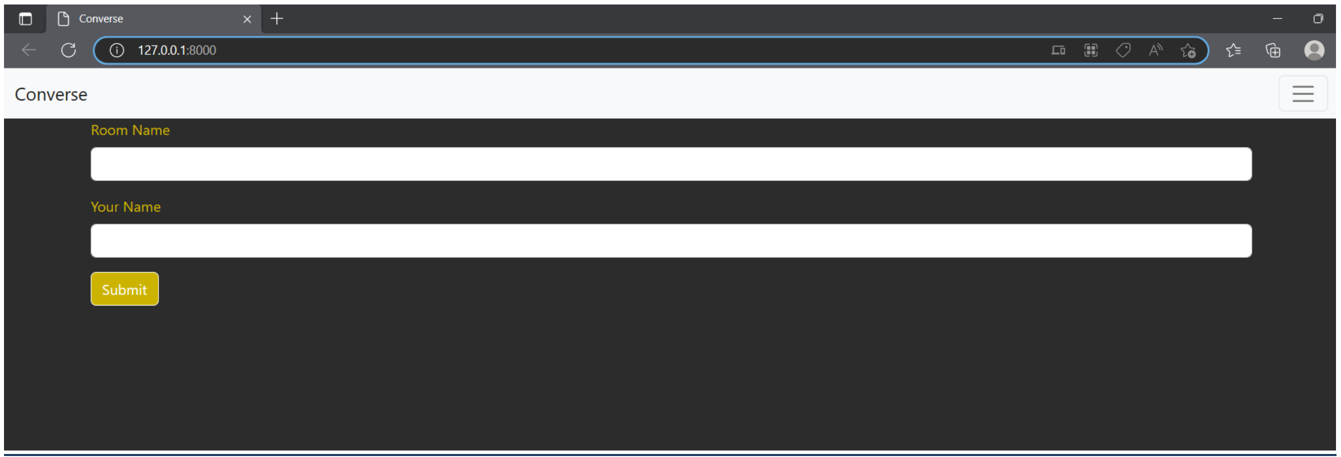
where are you?

Did you studied machine learning ?

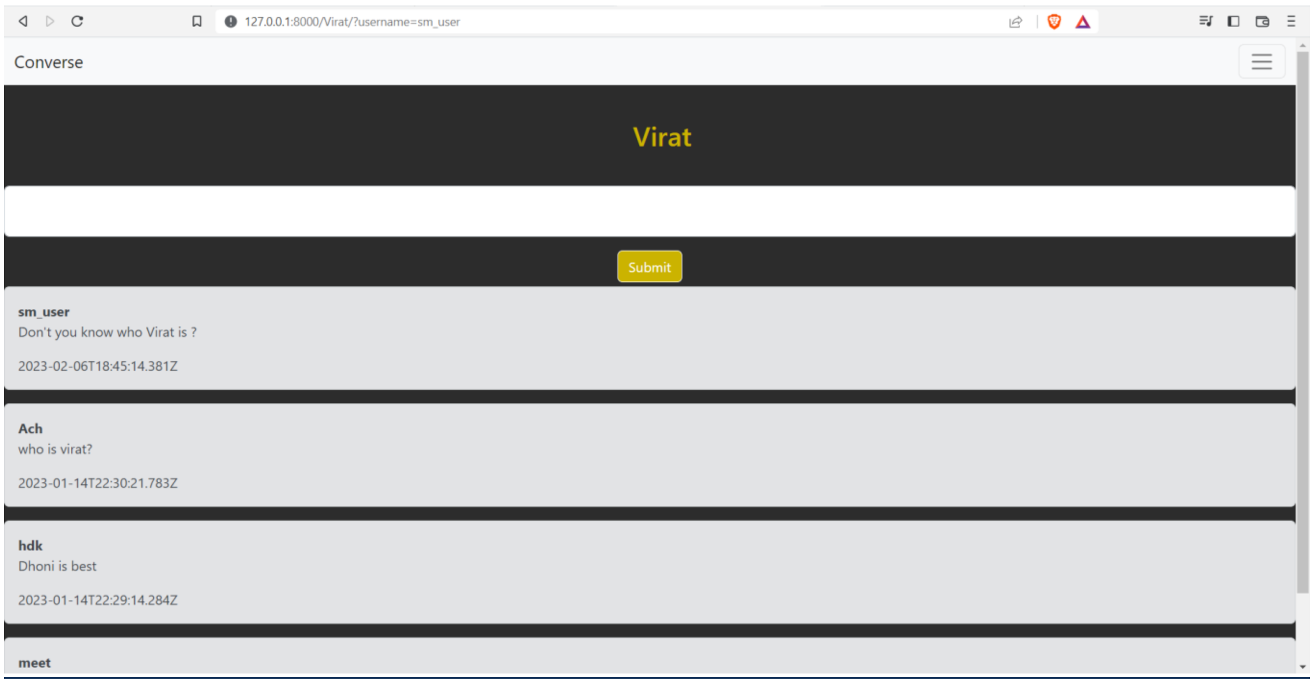
I've done my research on machine learning :)

Yes I've finished my online course

CHATROOM (room entry)



CHATROOM (conversations)



sm_user
Don't you know who Virat is ?

2023-02-06T18:45:14.381Z

Ach
who is virat?

2023-01-14T22:30:21.783Z

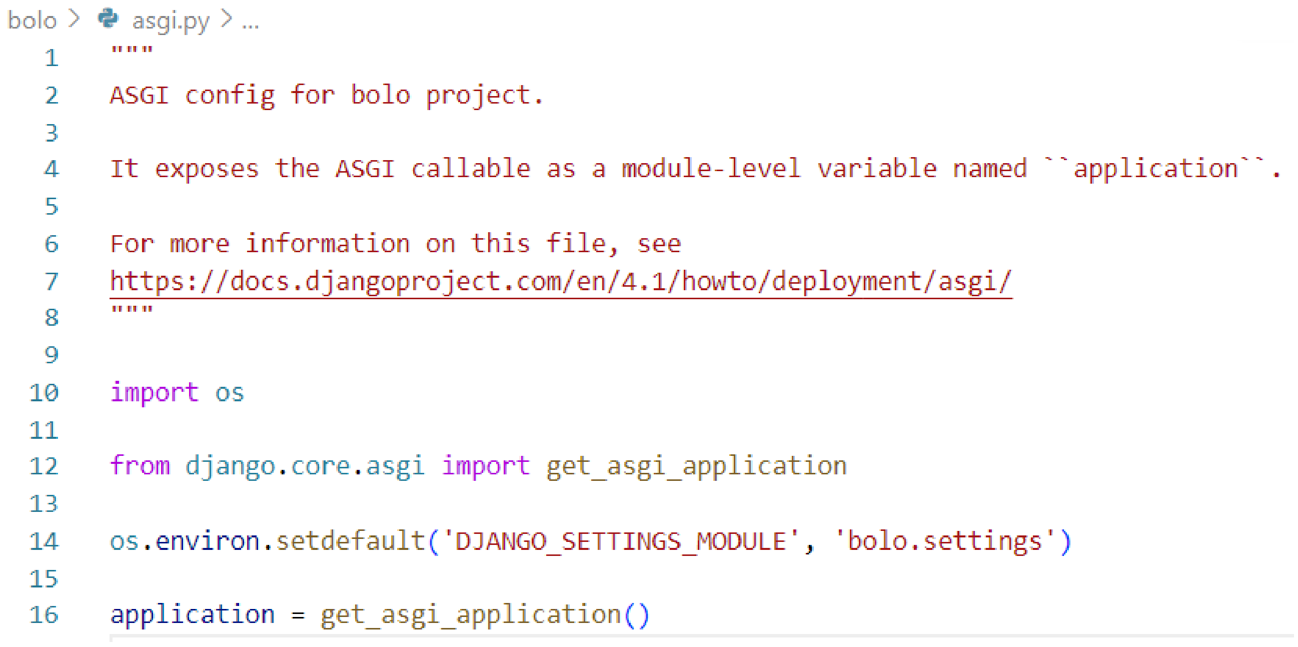
hdk
Dhoni is best

2023-01-14T22:29:14.284Z

meet

☯ Coding (Implementation) ☯

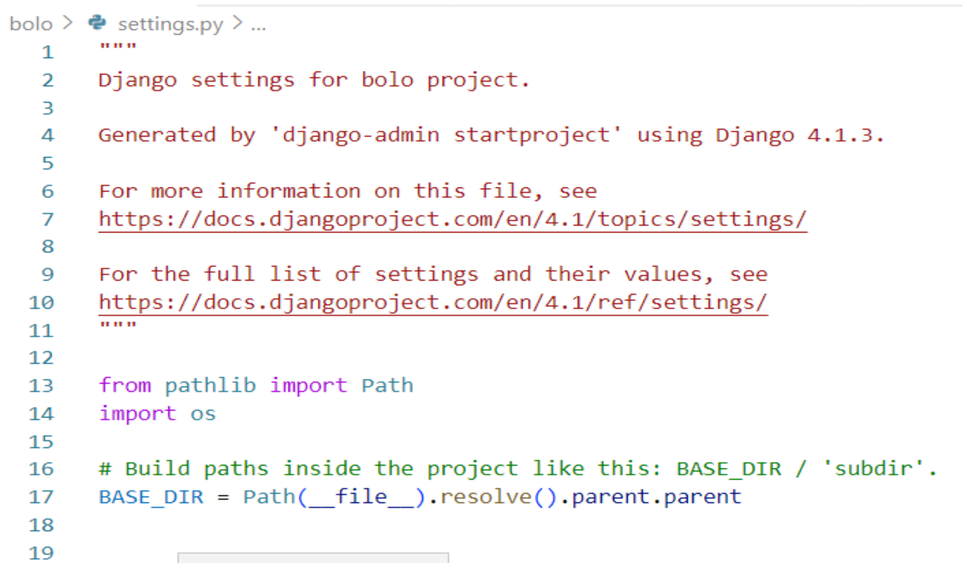
1.asgi.py



bolo > asgi.py > ...

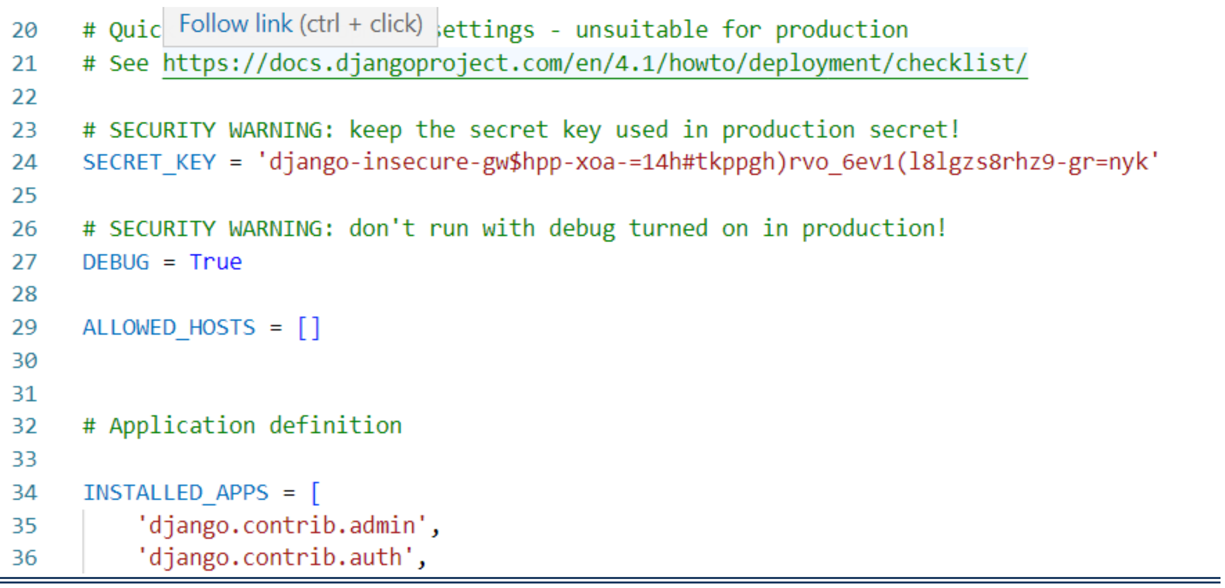
```python
"""
ASGI config for bolo project.

It exposes the ASGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.1/howto/deployment/asgi/
"""

import os

from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'bolo.settings')

application = get_asgi_application()
```

2.settings.py

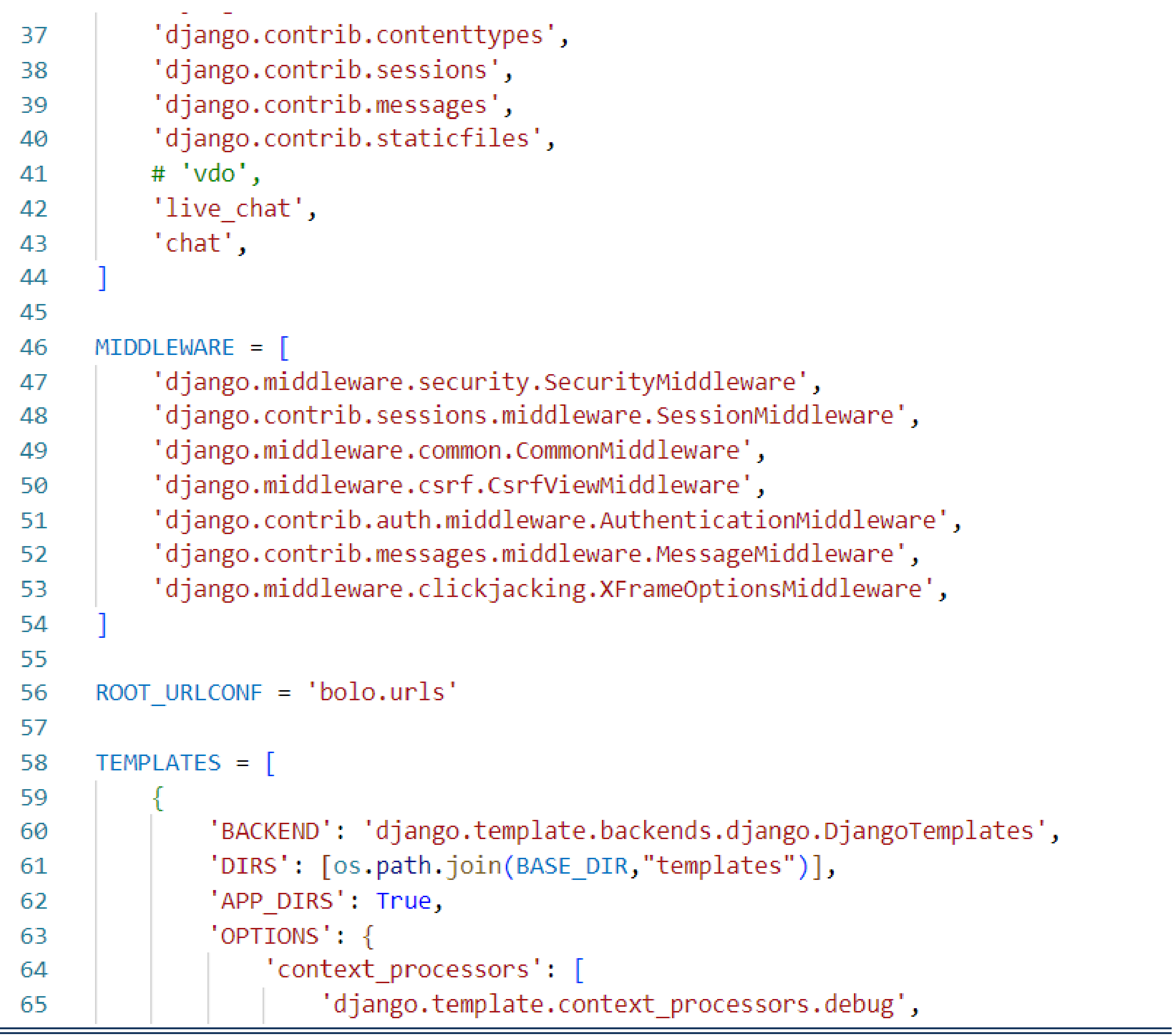


bolo > settings.py > ...

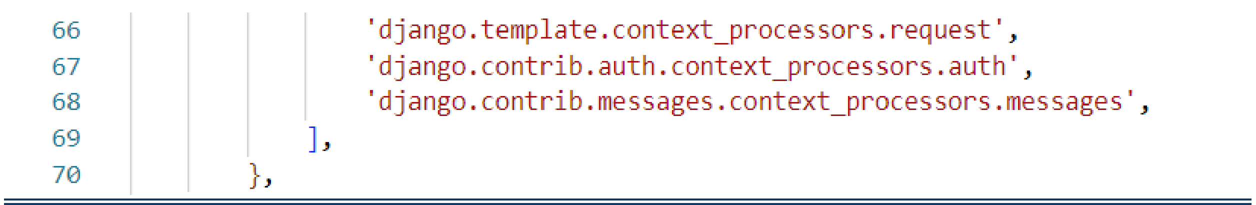
```python
"""
Django settings for bolo project.

Generated by 'django-admin startproject' using Django 4.1.3.

For more information on this file, see
https://docs.djangoproject.com/en/4.1/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/4.1/ref/settings/
"""

from pathlib import Path
import os

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent


```



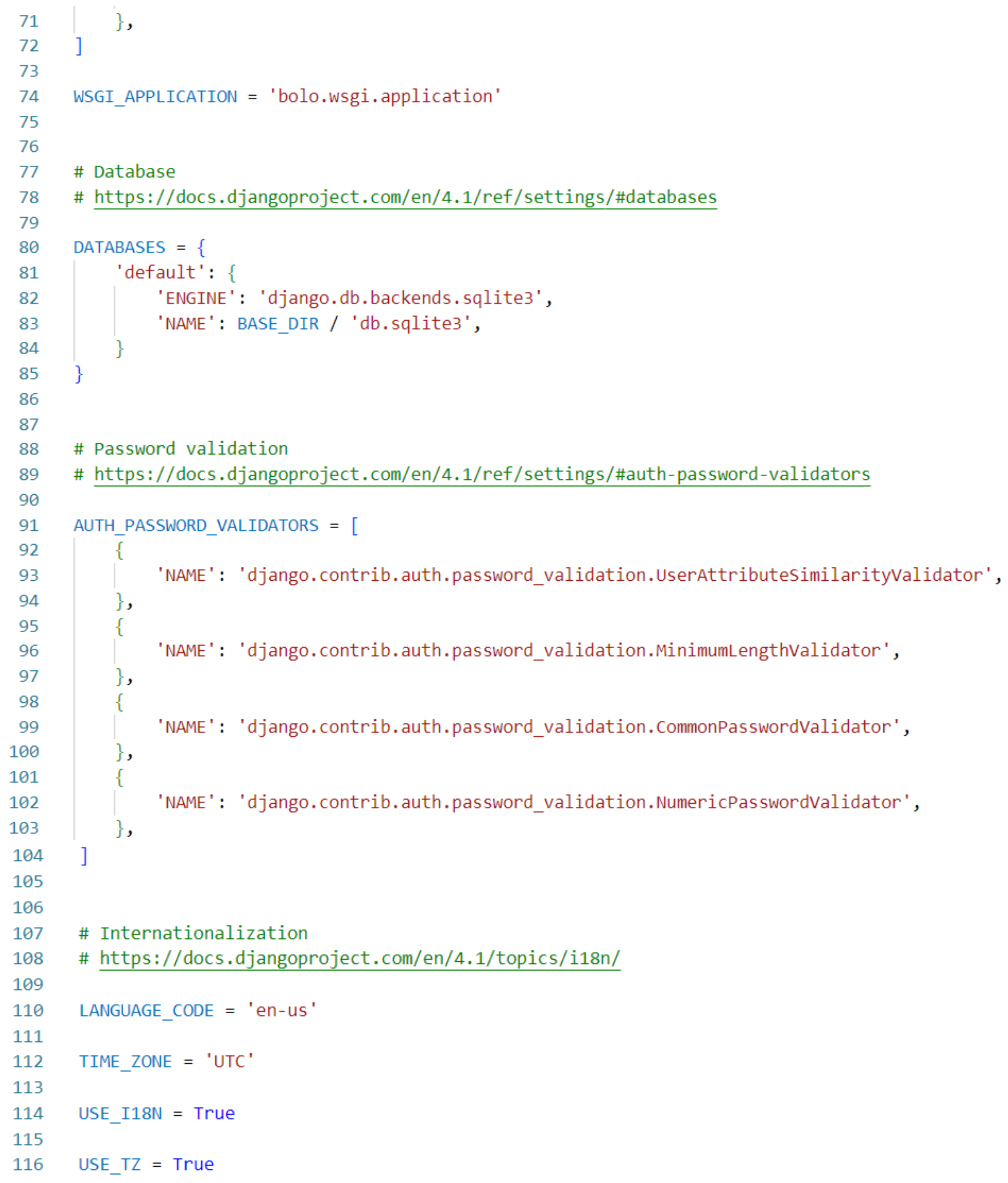
```python
# Quic Follow link (ctrl + click) ;ettings - unsuitable for production
# See https://docs.djangoproject.com/en/4.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'django-insecure-gw$hpp-xoa-=14h#tkppgh)rvo_6ev1(l8lgzs8rhz9-gr=nyk'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
```



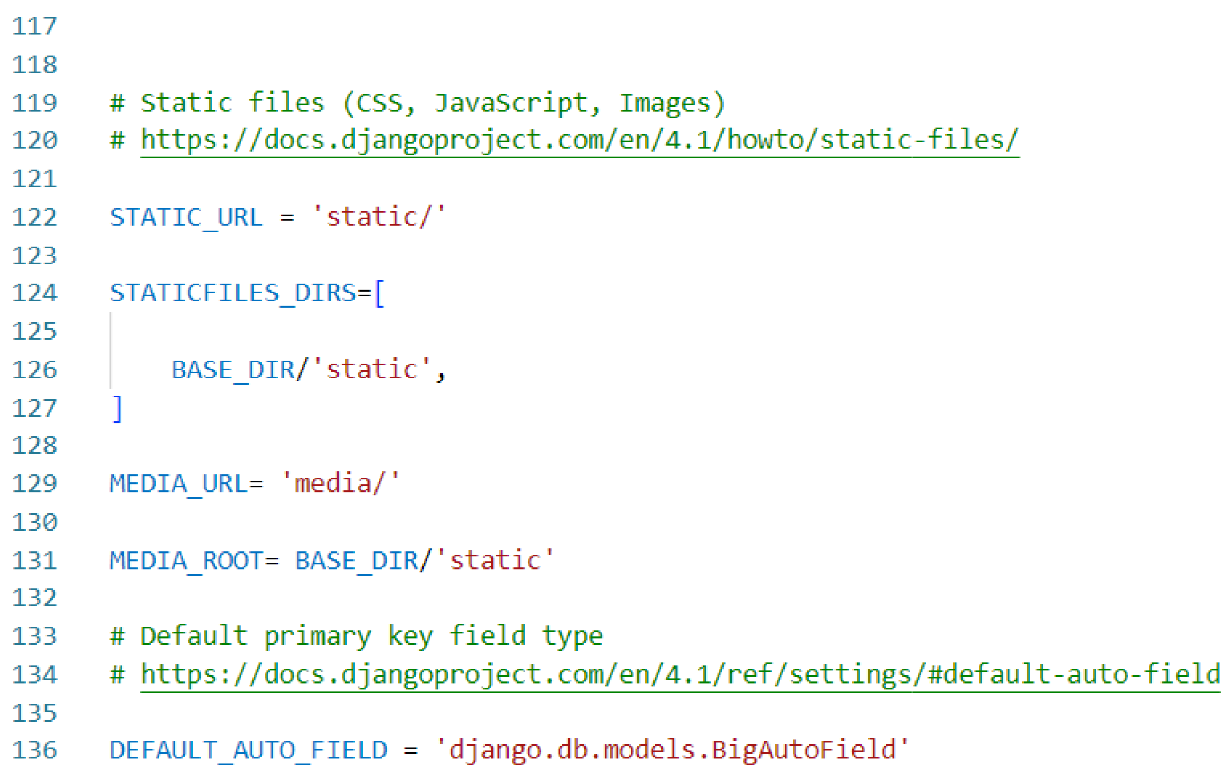
```python
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    # 'vdo',
    'live_chat',
    'chat',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'bolo.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,"templates")],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
```



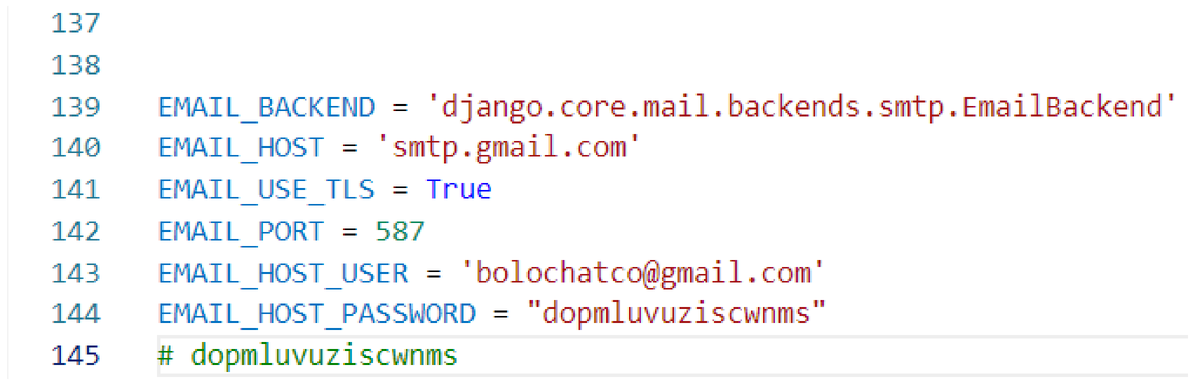
```python
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
```



```python
    },
]

WSGI_APPLICATION = 'bolo.wsgi.application'


# Database
# https://docs.djangoproject.com/en/4.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}


# Password validation
# https://docs.djangoproject.com/en/4.1/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/4.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True
```

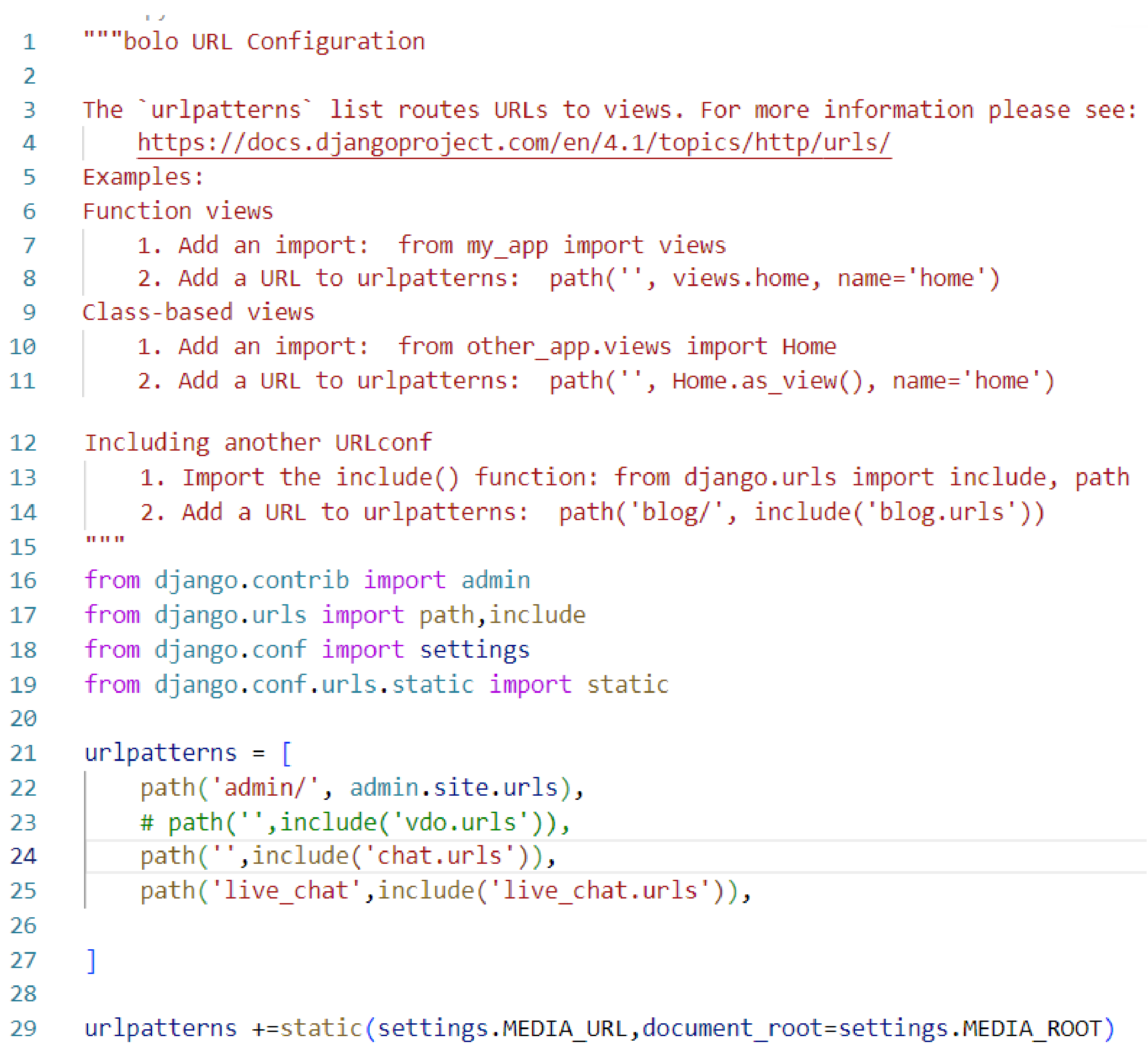


```python


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.1/howto/static-files/

STATIC_URL = 'static/'

STATICFILES_DIRS=[

    BASE_DIR/'static',
]

MEDIA_URL= 'media/'

MEDIA_ROOT= BASE_DIR/'static'

# Default primary key field type
# https://docs.djangoproject.com/en/4.1/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```



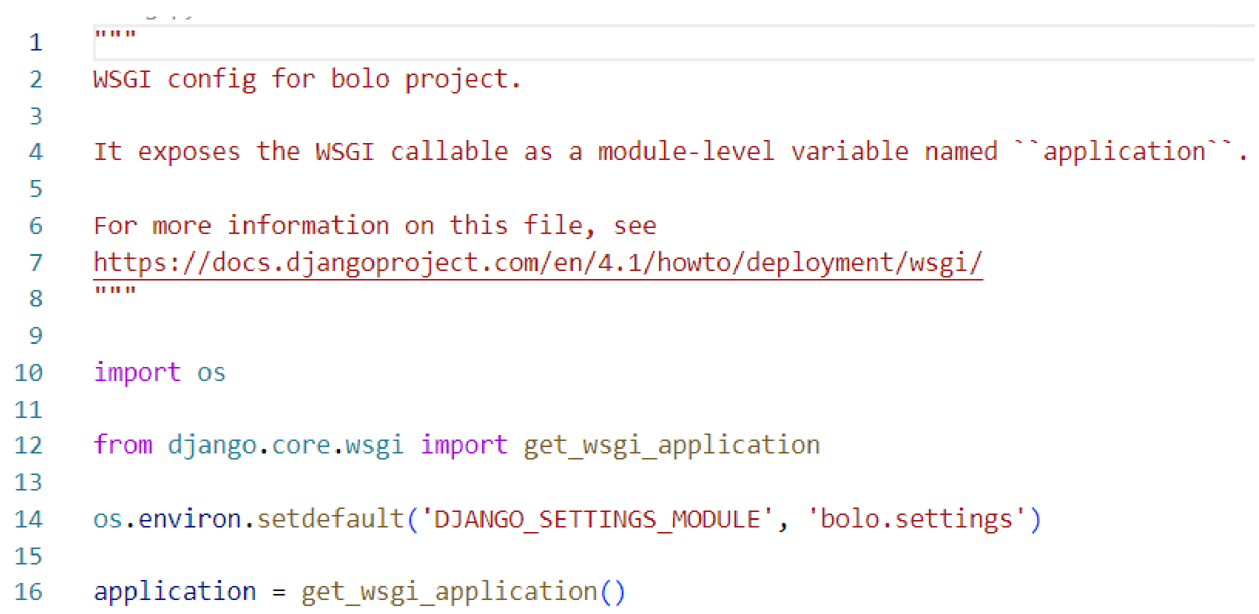
```python


EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_USE_TLS = True
EMAIL_PORT = 587
EMAIL_HOST_USER = 'bolochatco@gmail.com'
EMAIL_HOST_PASSWORD = "dopmluvuziscwnms"
# dopmluvuziscwnms
```

3.urls.py



```python
"""bolo URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/4.1/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path,include
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    # path('',include('vdo.urls')),
    path('',include('chat.urls')),
    path('live_chat',include('live_chat.urls')),

]

urlpatterns +=static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)
```

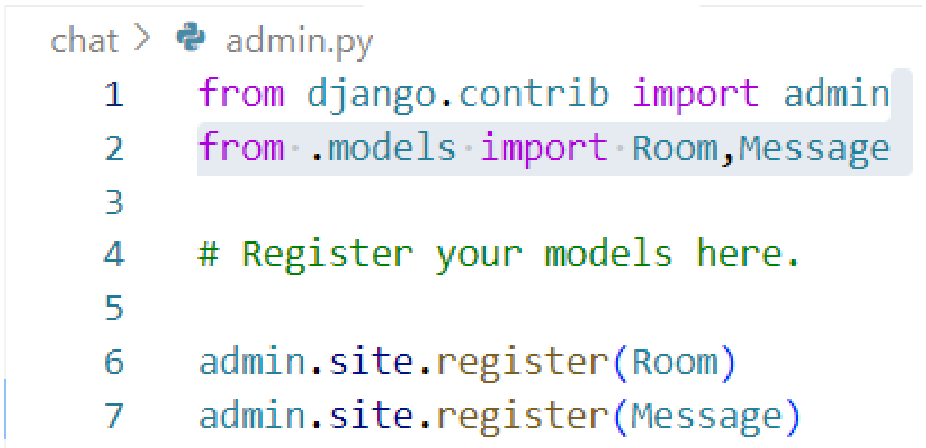
4.wsgi.py:



```python
"""
WSGI config for bolo project.

It exposes the WSGI callable as a module-level variable named ``application``.

For more information on this file, see
https://docs.djangoproject.com/en/4.1/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'bolo.settings')

application = get_wsgi_application()
```

➢ Chat:

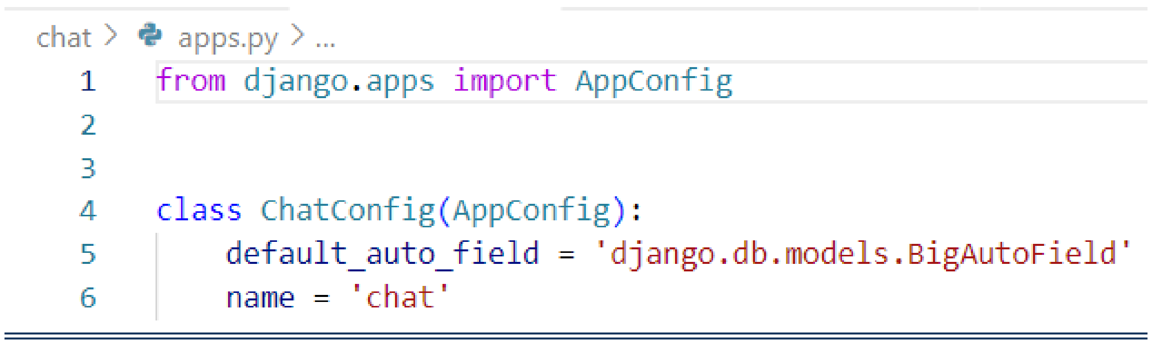
1.admin.py:



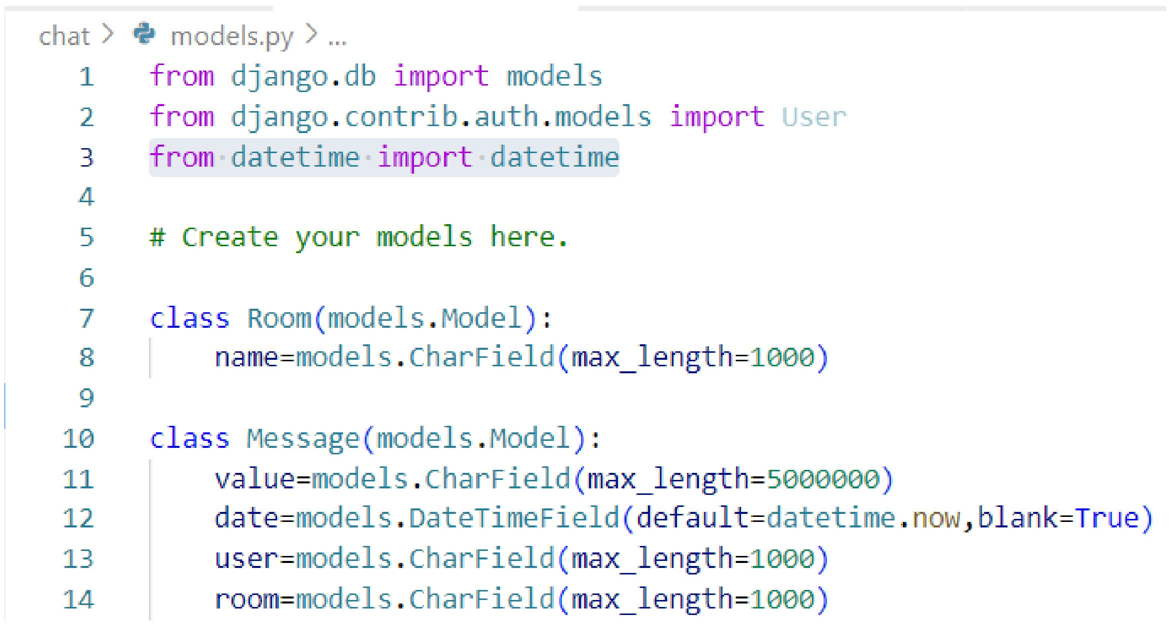
chat > admin.py

```python
from django.contrib import admin
from .models import Room,Message

# Register your models here.

admin.site.register(Room)
admin.site.register(Message)
```

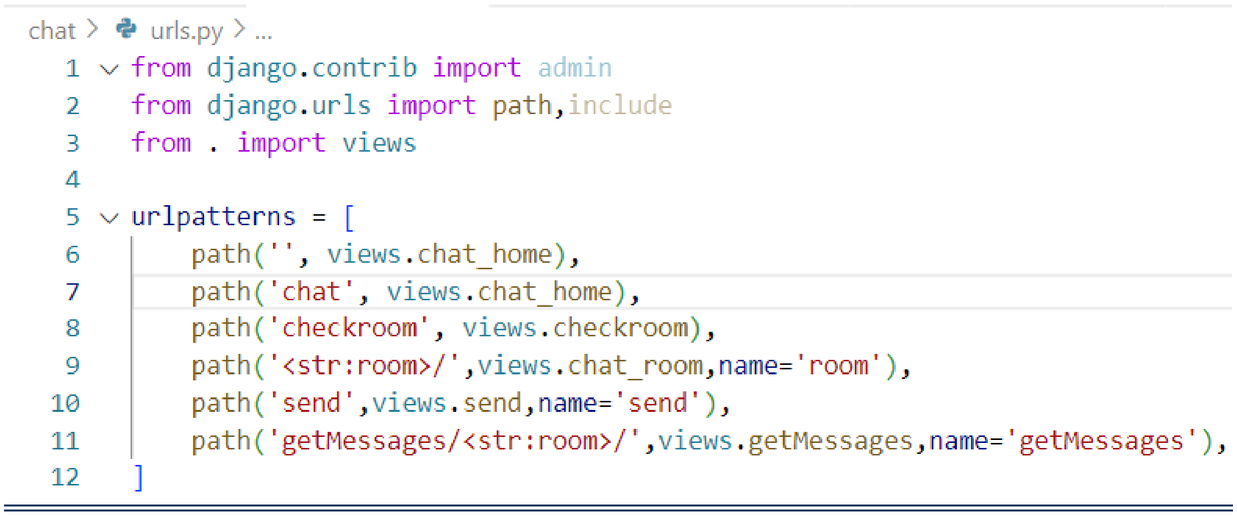
2.apps.py:


chat > apps.py > ...
```python
from django.apps import AppConfig


class ChatConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'chat'
```

3.models.py:

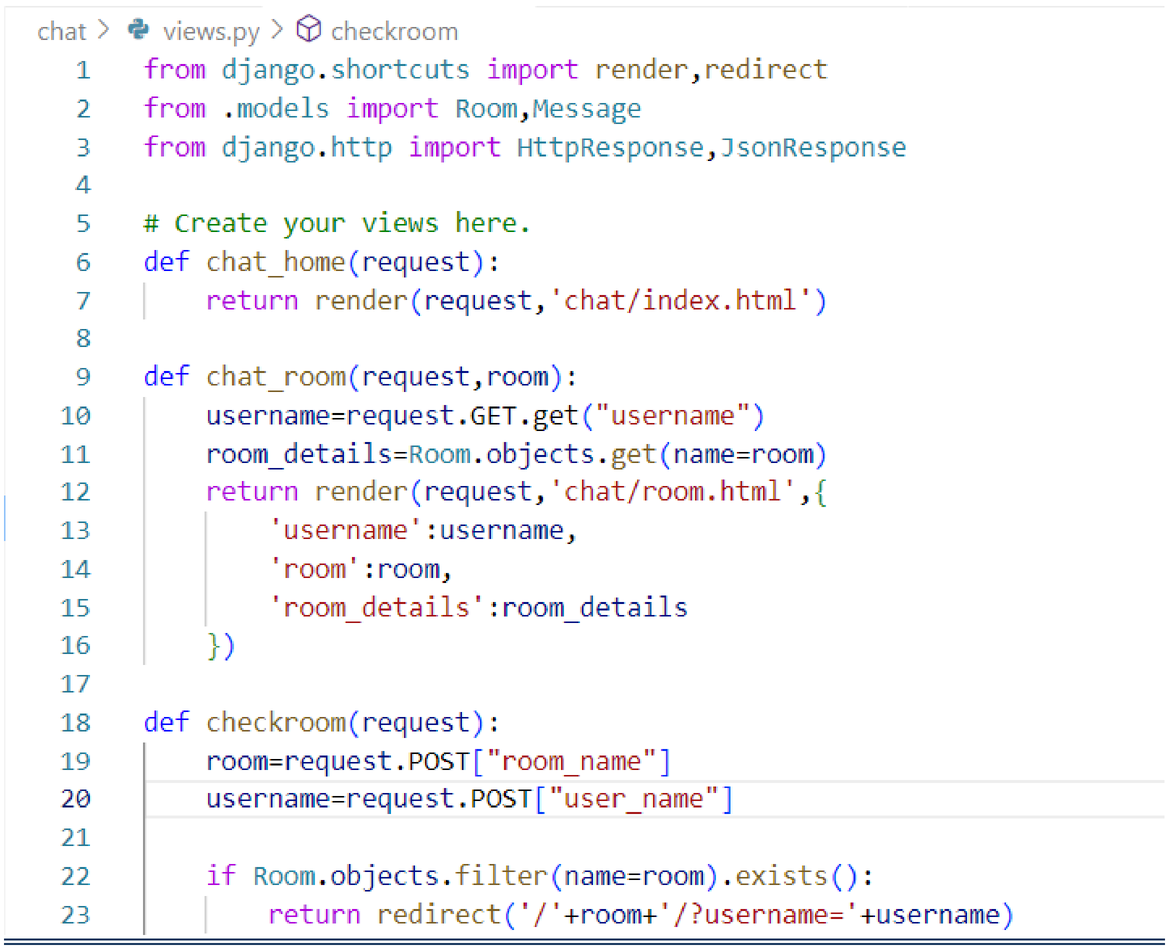

chat > models.py > ...
```python
from django.db import models
from django.contrib.auth.models import User
from datetime import datetime

# Create your models here.

class Room(models.Model):
    name=models.CharField(max_length=1000)

class Message(models.Model):
    value=models.CharField(max_length=5000000)
    date=models.DateTimeField(default=datetime.now,blank=True)
    user=models.CharField(max_length=1000)
    room=models.CharField(max_length=1000)
```

4.urls.py:

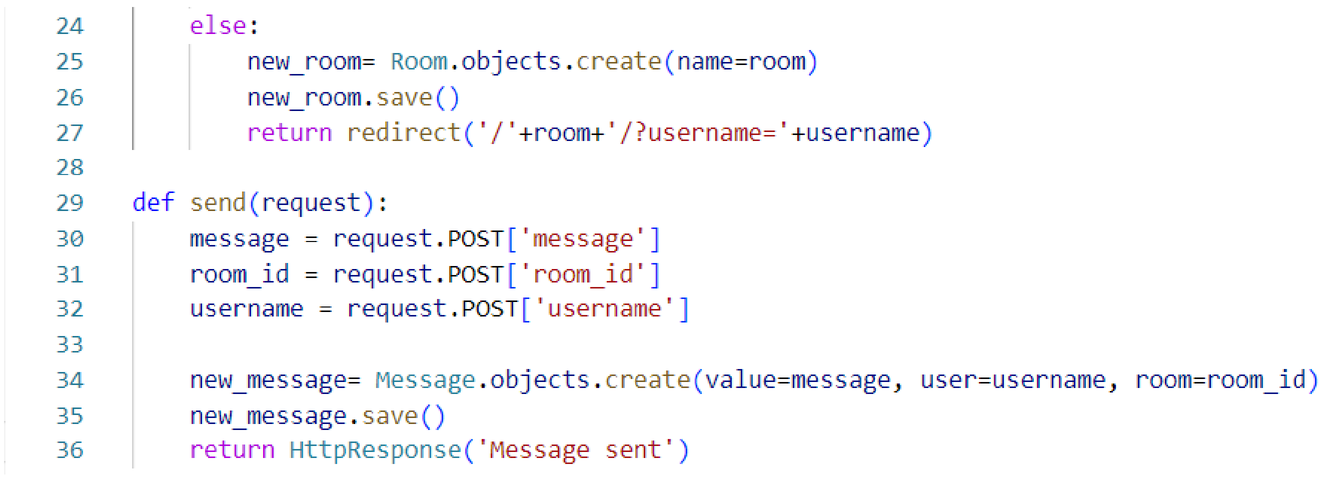

chat > urls.py > ...

```python
from django.contrib import admin
from django.urls import path,include
from . import views

urlpatterns = [
    path('', views.chat_home),
    path('chat', views.chat_home),
    path('checkroom', views.checkroom),
    path('<str:room>/',views.chat_room,name='room'),
    path('send',views.send,name='send'),
    path('getMessages/<str:room>/',views.getMessages,name='getMessages'),
]
```

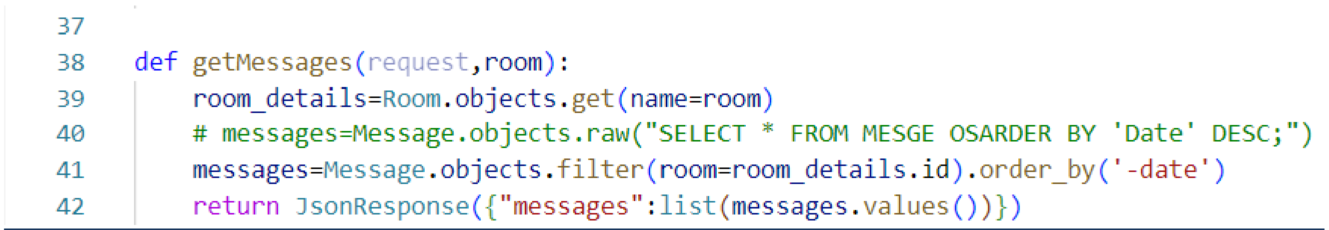
5.views.py:


chat > views.py > checkroom

```python
from django.shortcuts import render,redirect
from .models import Room,Message
from django.http import HttpResponse,JsonResponse

# Create your views here.
def chat_home(request):
    return render(request,'chat/index.html')

def chat_room(request,room):
    username=request.GET.get("username")
    room_details=Room.objects.get(name=room)
    return render(request,'chat/room.html',{
        'username':username,
        'room':room,
        'room_details':room_details
    })

def checkroom(request):
    room=request.POST["room_name"]
    username=request.POST["user_name"]

    if Room.objects.filter(name=room).exists():
        return redirect('/'+room+'/?username='+username)
```



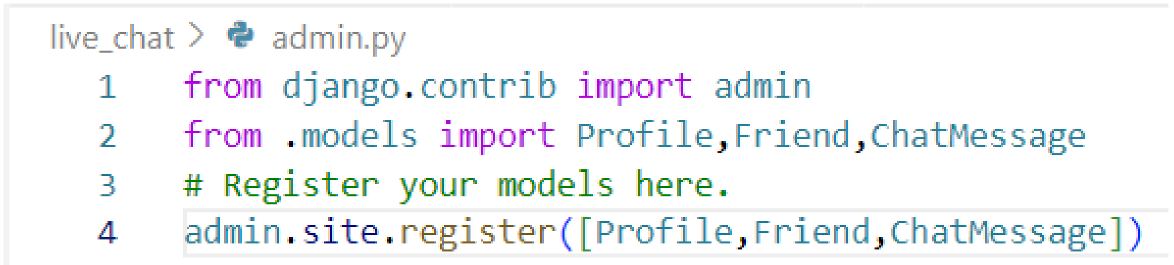
```python
    else:
        new_room= Room.objects.create(name=room)
        new_room.save()
        return redirect('/'+room+'/?username='+username)

def send(request):
    message = request.POST['message']
    room_id = request.POST['room_id']
    username = request.POST['username']

    new_message= Message.objects.create(value=message, user=username, room=room_id)
    new_message.save()
    return HttpResponse('Message sent')
```



```python

def getMessages(request,room):
    room_details=Room.objects.get(name=room)
    # messages=Message.objects.raw("SELECT * FROM MESGE OSARDER BY 'Date' DESC;")
    messages=Message.objects.filter(room=room_details.id).order_by('-date')
    return JsonResponse({"messages":list(messages.values())})
```

➢ Live_chat:

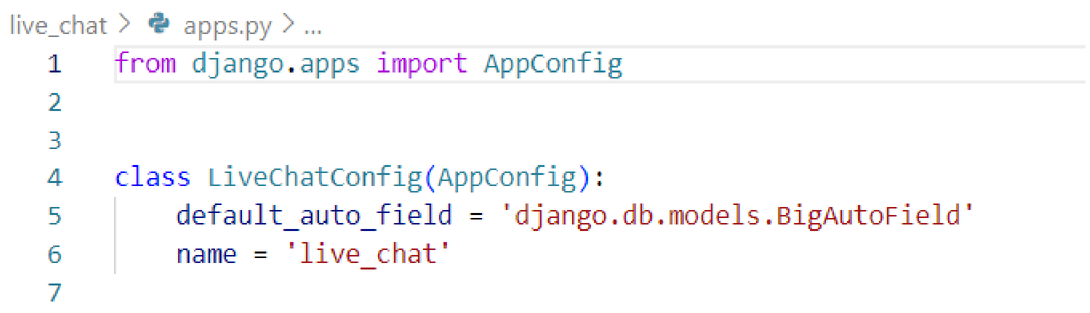
1.admin.py



live_chat > admin.py

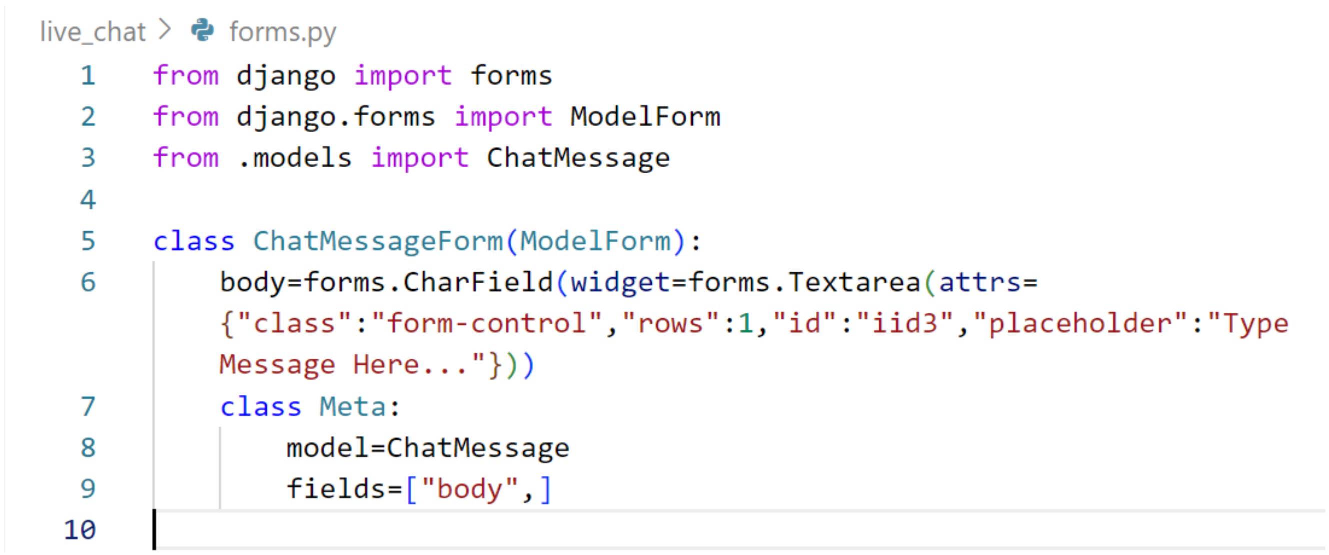
```python
from django.contrib import admin
from .models import Profile,Friend,ChatMessage
# Register your models here.
admin.site.register([Profile,Friend,ChatMessage])
```

2.apps.py



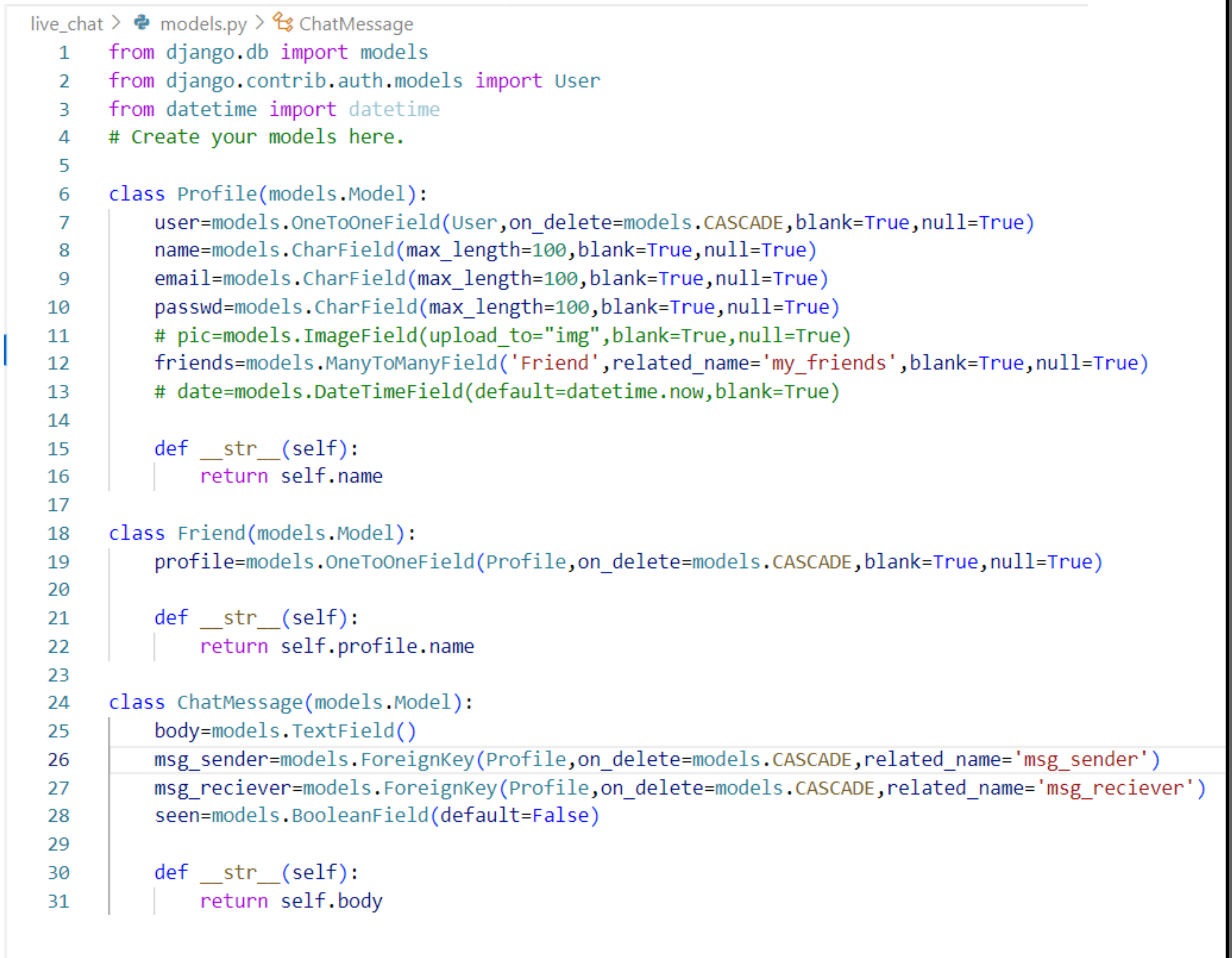
```python
from django.apps import AppConfig


class LiveChatConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'live_chat'

```

3.forms.py



```python
from django import forms
from django.forms import ModelForm
from .models import ChatMessage

class ChatMessageForm(ModelForm):
    body=forms.CharField(widget=forms.Textarea(attrs={"class":"form-control","rows":1,"id":"iid3","placeholder":"Type Message Here..."}))
    class Meta:
        model=ChatMessage
        fields=["body",]

```

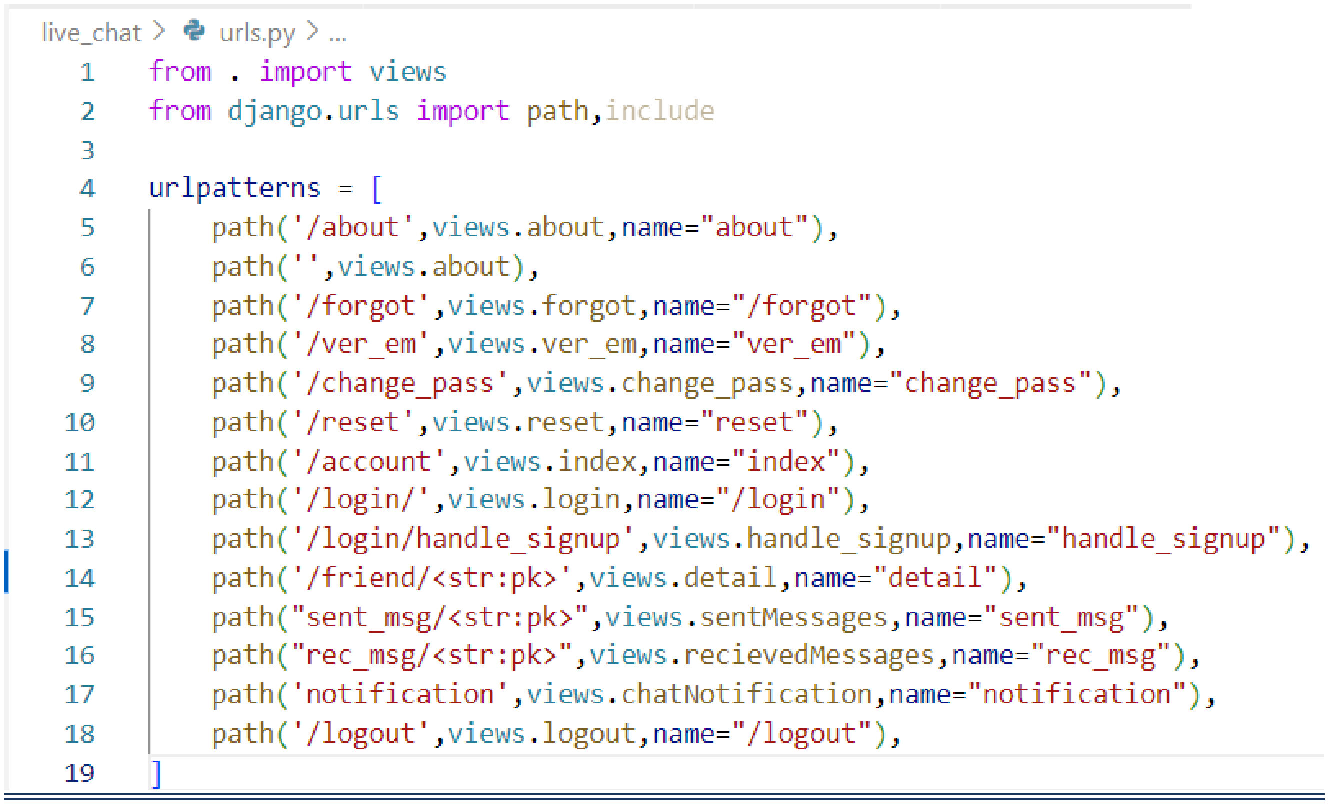
4.models.py



live_chat > models.py > ChatMessage

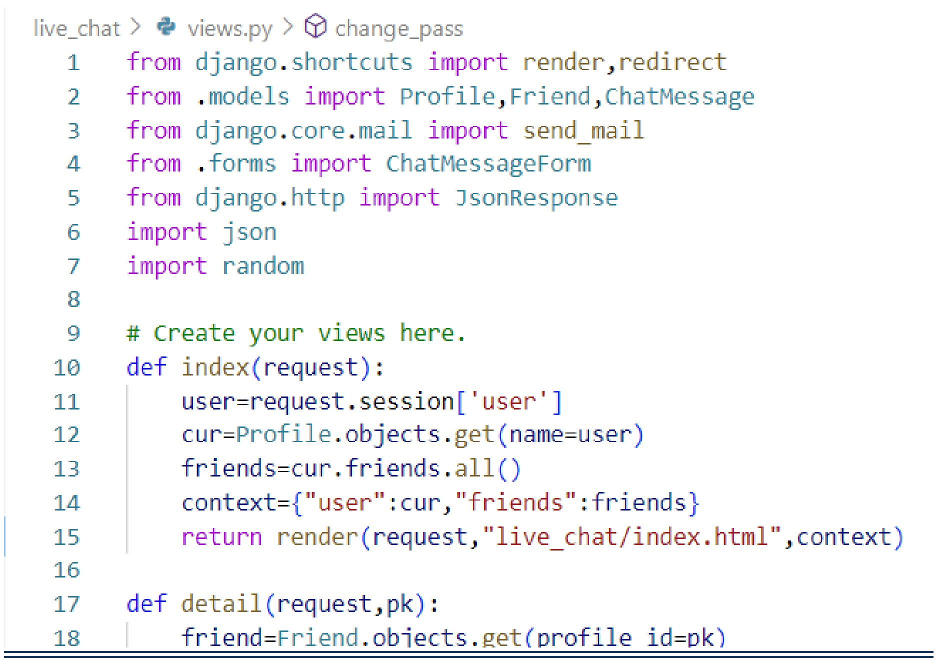
```python
from django.db import models
from django.contrib.auth.models import User
from datetime import datetime
# Create your models here.

class Profile(models.Model):
    user=models.OneToOneField(User,on_delete=models.CASCADE,blank=True,null=True)
    name=models.CharField(max_length=100,blank=True,null=True)
    email=models.CharField(max_length=100,blank=True,null=True)
    passwd=models.CharField(max_length=100,blank=True,null=True)
    # pic=models.ImageField(upload_to="img",blank=True,null=True)
    friends=models.ManyToManyField('Friend',related_name='my_friends',blank=True,null=True)
    # date=models.DateTimeField(default=datetime.now,blank=True)

    def __str__(self):
        return self.name

class Friend(models.Model):
    profile=models.OneToOneField(Profile,on_delete=models.CASCADE,blank=True,null=True)

    def __str__(self):
        return self.profile.name

class ChatMessage(models.Model):
    body=models.TextField()
    msg_sender=models.ForeignKey(Profile,on_delete=models.CASCADE,related_name='msg_sender')
    msg_reciever=models.ForeignKey(Profile,on_delete=models.CASCADE,related_name='msg_reciever')
    seen=models.BooleanField(default=False)

    def __str__(self):
        return self.body
```

5.urls.py

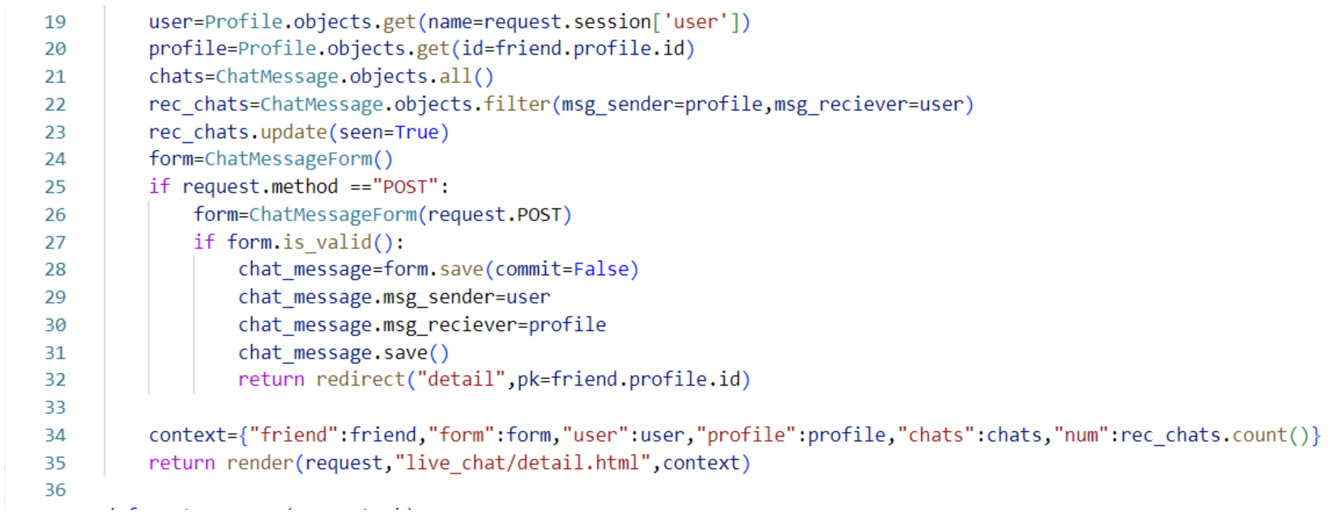


```python
from . import views
from django.urls import path,include

urlpatterns = [
    path('/about',views.about,name="about"),
    path('',views.about),
    path('/forgot',views.forgot,name="/forgot"),
    path('/ver_em',views.ver_em,name="ver_em"),
    path('/change_pass',views.change_pass,name="change_pass"),
    path('/reset',views.reset,name="reset"),
    path('/account',views.index,name="index"),
    path('/login/',views.login,name="/login"),
    path('/login/handle_signup',views.handle_signup,name="handle_signup"),
    path('/friend/<str:pk>',views.detail,name="detail"),
    path("sent_msg/<str:pk>",views.sentMessages,name="sent_msg"),
    path("rec_msg/<str:pk>",views.recievedMessages,name="rec_msg"),
    path('notification',views.chatNotification,name="notification"),
    path('/logout',views.logout,name="/logout"),
]
```

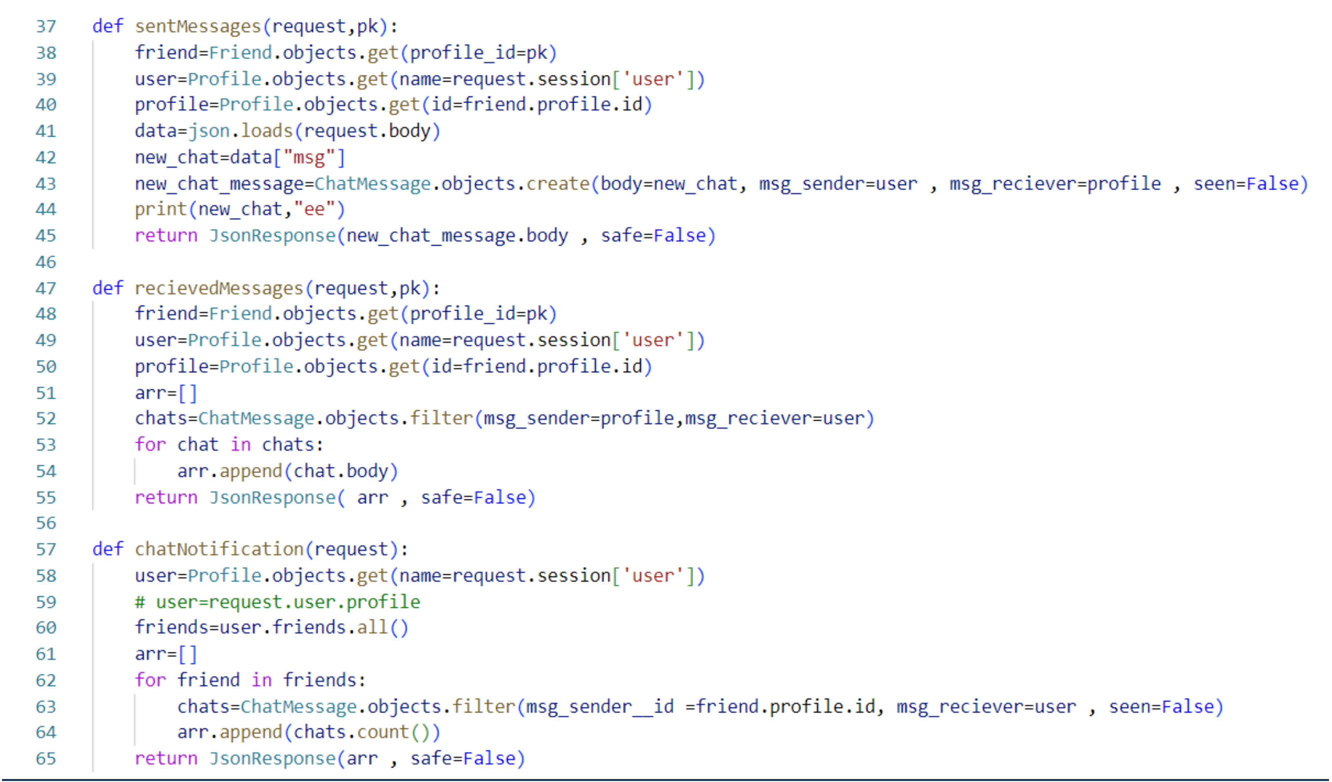
6.views.py



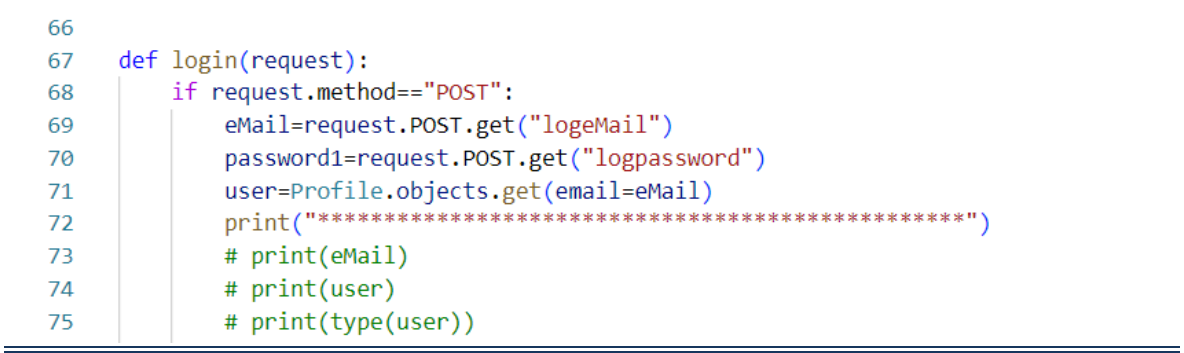
```python
from django.shortcuts import render,redirect
from .models import Profile,Friend,ChatMessage
from django.core.mail import send_mail
from .forms import ChatMessageForm
from django.http import JsonResponse
import json
import random

# Create your views here.
def index(request):
    user=request.session['user']
    cur=Profile.objects.get(name=user)
    friends=cur.friends.all()
    context={"user":cur,"friends":friends}
    return render(request,"live_chat/index.html",context)

def detail(request,pk):
    friend=Friend.objects.get(profile_id=pk)
```



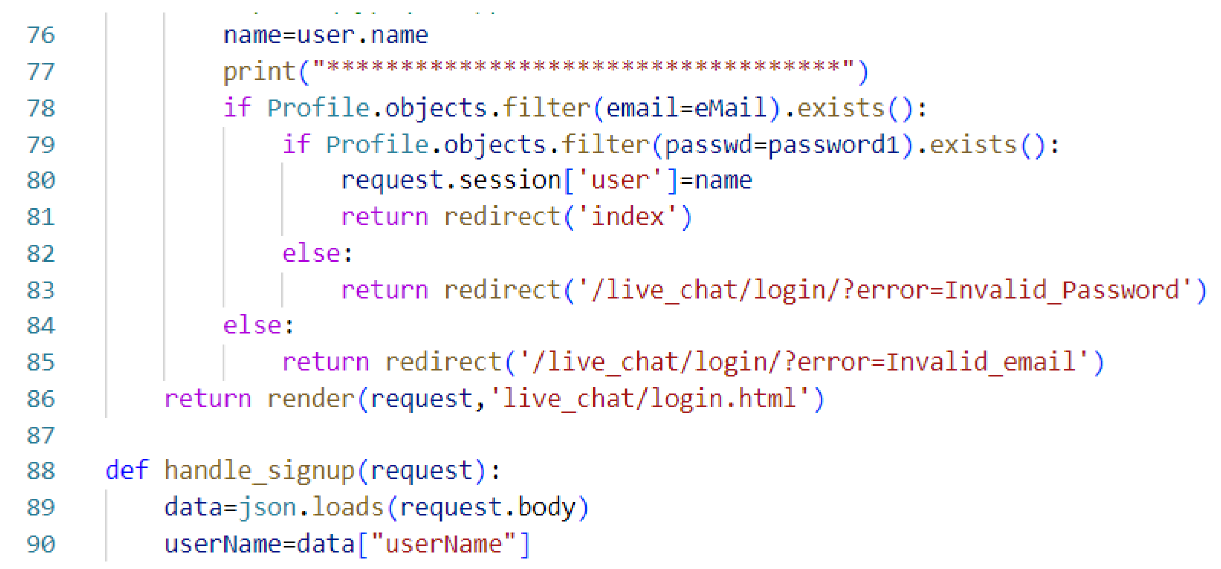
```python
    user=Profile.objects.get(name=request.session['user'])
    profile=Profile.objects.get(id=friend.profile.id)
    chats=ChatMessage.objects.all()
    rec_chats=ChatMessage.objects.filter(msg_sender=profile,msg_reciever=user)
    rec_chats.update(seen=True)
    form=ChatMessageForm()
    if request.method =="POST":
        form=ChatMessageForm(request.POST)
        if form.is_valid():
            chat_message=form.save(commit=False)
            chat_message.msg_sender=user
            chat_message.msg_reciever=profile
            chat_message.save()
            return redirect("detail",pk=friend.profile.id)

    context={"friend":friend,"form":form,"user":user,"profile":profile,"chats":chats,"num":rec_chats.count()}
    return render(request,"live_chat/detail.html",context)

```



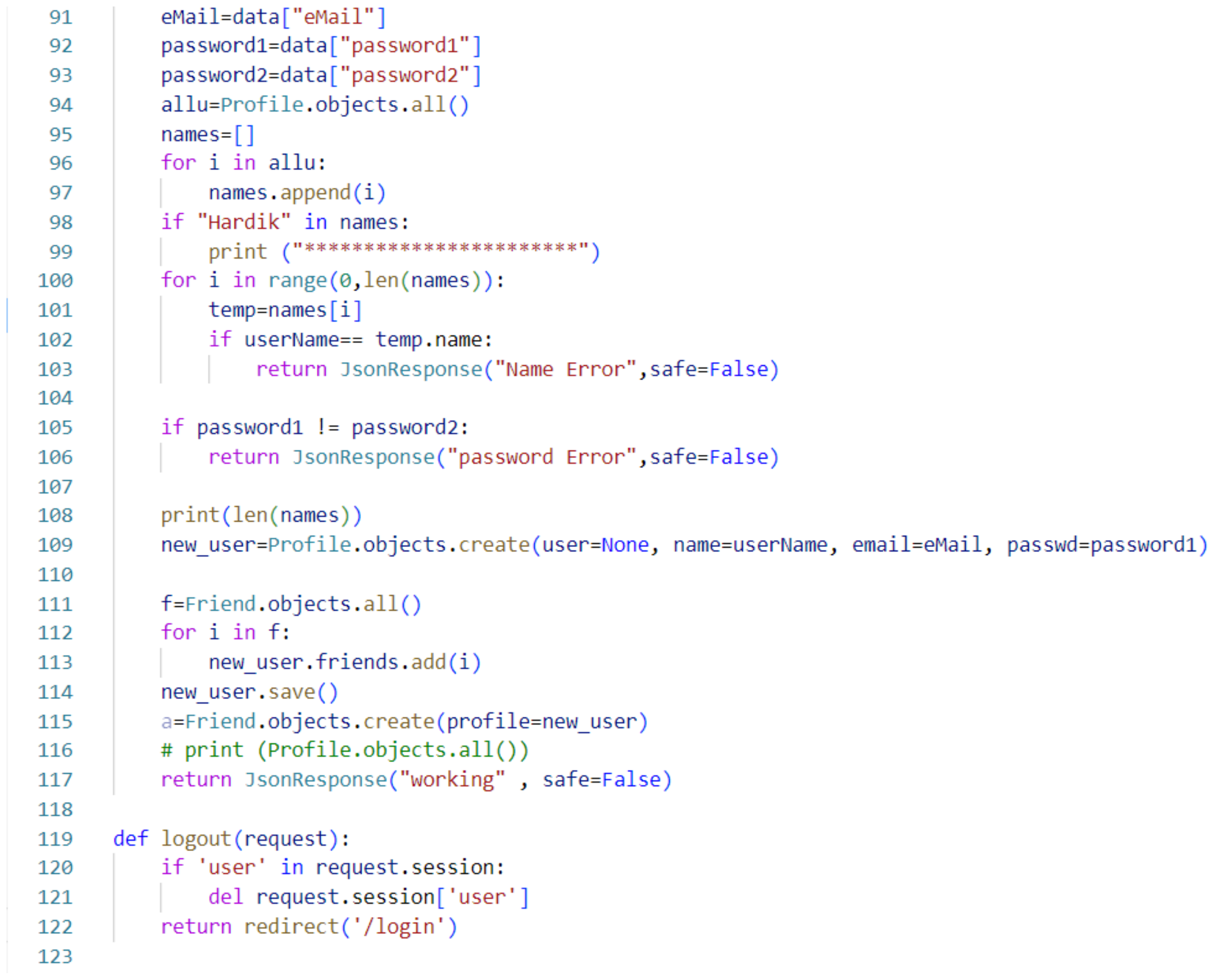
```python
def sentMessages(request,pk):
    friend=Friend.objects.get(profile_id=pk)
    user=Profile.objects.get(name=request.session['user'])
    profile=Profile.objects.get(id=friend.profile.id)
    data=json.loads(request.body)
    new_chat=data["msg"]
    new_chat_message=ChatMessage.objects.create(body=new_chat, msg_sender=user , msg_reciever=profile , seen=False)
    print(new_chat,"ee")
    return JsonResponse(new_chat_message.body , safe=False)

def recievedMessages(request,pk):
    friend=Friend.objects.get(profile_id=pk)
    user=Profile.objects.get(name=request.session['user'])
    profile=Profile.objects.get(id=friend.profile.id)
    arr=[]
    chats=ChatMessage.objects.filter(msg_sender=profile,msg_reciever=user)
    for chat in chats:
        arr.append(chat.body)
    return JsonResponse( arr , safe=False)

def chatNotification(request):
    user=Profile.objects.get(name=request.session['user'])
    # user=request.user.profile
    friends=user.friends.all()
    arr=[]
    for friend in friends:
        chats=ChatMessage.objects.filter(msg_sender__id =friend.profile.id, msg_reciever=user , seen=False)
        arr.append(chats.count())
    return JsonResponse(arr , safe=False)
```



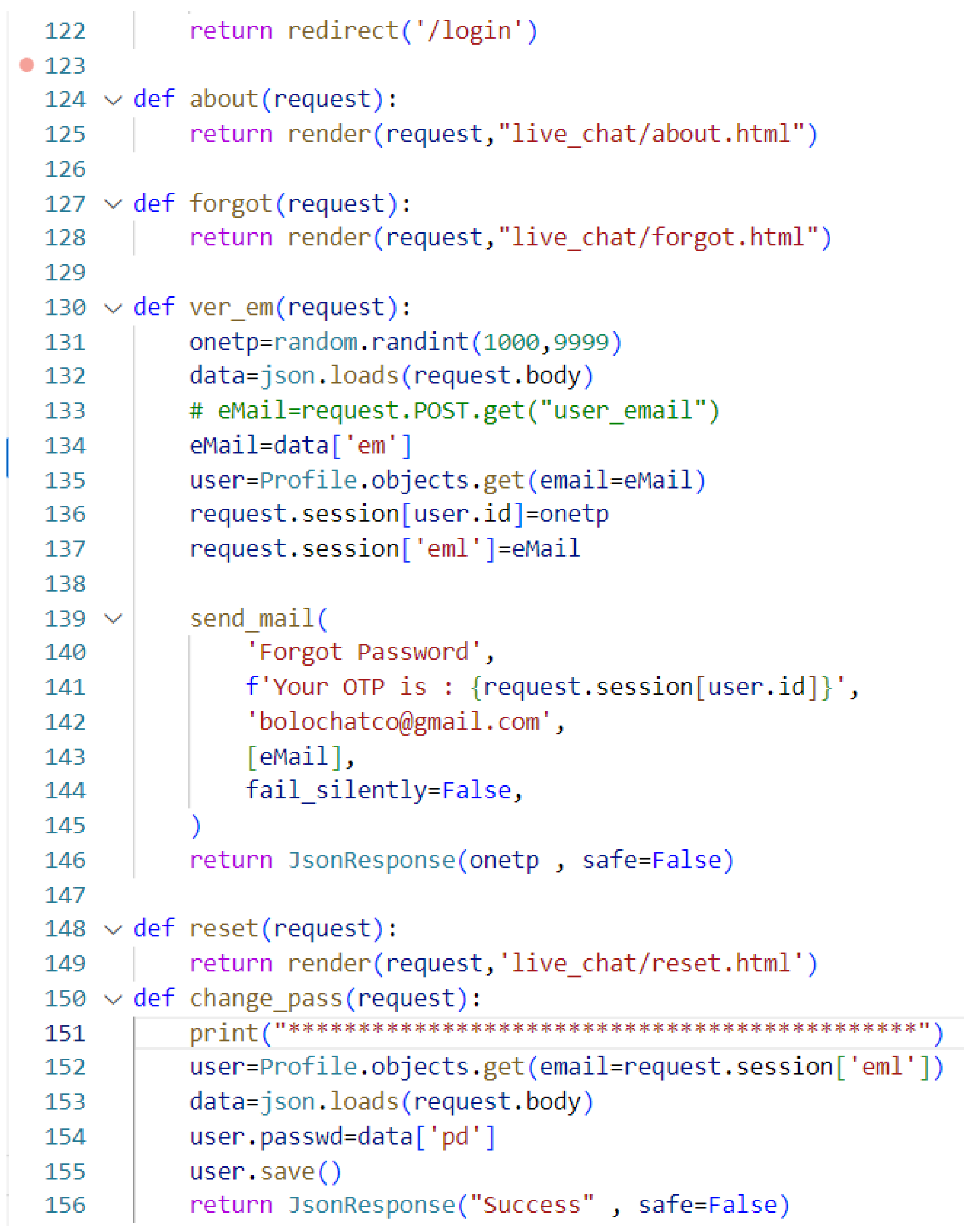
```python

def login(request):
    if request.method=="POST":
        eMail=request.POST.get("logeMail")
        password1=request.POST.get("logpassword")
        user=Profile.objects.get(email=eMail)
        print("**************************************************")
        # print(eMail)
        # print(user)
        # print(type(user))
```



```python
        name=user.name
        print("************************************")
        if Profile.objects.filter(email=eMail).exists():
            if Profile.objects.filter(passwd=password1).exists():
                request.session['user']=name
                return redirect('index')
            else:
                return redirect('/live_chat/login/?error=Invalid_Password')
        else:
            return redirect('/live_chat/login/?error=Invalid_email')
    return render(request,'live_chat/login.html')

def handle_signup(request):
    data=json.loads(request.body)
    userName=data["userName"]
```



```python
    eMail=data["eMail"]
    password1=data["password1"]
    password2=data["password2"]
    allu=Profile.objects.all()
    names=[]
    for i in allu:
        names.append(i)
    if "Hardik" in names:
        print ("***********************")
    for i in range(0,len(names)):
        temp=names[i]
        if userName== temp.name:
            return JsonResponse("Name Error",safe=False)

    if password1 != password2:
        return JsonResponse("password Error",safe=False)

    print(len(names))
    new_user=Profile.objects.create(user=None, name=userName, email=eMail, passwd=password1)

    f=Friend.objects.all()
    for i in f:
        new_user.friends.add(i)
    new_user.save()
    a=Friend.objects.create(profile=new_user)
    # print (Profile.objects.all())
    return JsonResponse("working" , safe=False)

def logout(request):
    if 'user' in request.session:
        del request.session['user']
    return redirect('/login')
```



```python
    return redirect('/login')

def about(request):
    return render(request,"live_chat/about.html")

def forgot(request):
    return render(request,"live_chat/forgot.html")

def ver_em(request):
    onetp=random.randint(1000,9999)
    data=json.loads(request.body)
    # eMail=request.POST.get("user_email")
    eMail=data['em']
    user=Profile.objects.get(email=eMail)
    request.session[user.id]=onetp
    request.session['eml']=eMail

    send_mail(
        'Forgot Password',
        f'Your OTP is : {request.session[user.id]}',
        'bolochatco@gmail.com',
        [eMail],
        fail_silently=False,
    )
    return JsonResponse(onetp , safe=False)

def reset(request):
    return render(request,'live_chat/reset.html')
def change_pass(request):
    print("*********************************************")
    user=Profile.objects.get(email=request.session['eml'])
    data=json.loads(request.body)
    user.passwd=data['pd']
    user.save()
    return JsonResponse("Success" , safe=False)
```

Templates

➔ Chat

Base.html

```
<!DOCTYPE html>
{% load static %}
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Converse</title>
    <link rel="stylesheet" type='text/css' media='screen' href="{%static
'styles/main.css' %}">

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLwZJEdK2Kadq2F9CUG65"
crossorigin="anonymous">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
kenU1KFdBIe4zVF0s0G1M5b4hcpxyD9F7jL+jjXkk+Q2h455rYXK/7HAuoJl+0I4"
crossorigin="anonymous"></script>
<script  src="https://code.jquery.com/jquery-3.1.1.min.js"   integrity="sha256-
hVVnYaiADRTO2PzUGmuLJr8BLUSjGIZsDYGmIJLv2b8="   crossorigin="anonymous"></script>
<style>
  #inout{
    width: 99px;
    background-color: #2c2c2c;
    color: #cbb300;
    text-decoration: none;
    height: 40px;
    display: flex;
    justify-content: center;
    align-items: center;
    border: 2px solid white;
    border-radius: 30px;
  }
</style>
</head>

<body>
    <nav class="navbar bg-light sticky-top">
        <div class="container-fluid">
```

```
          <a class="navbar-brand" href="#">Converse</a>
          <button class="navbar-toggler" type="button" data-bs-toggle="offcanvas"
data-bs-target="#offcanvasNavbar" aria-controls="offcanvasNavbar">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="offcanvas offcanvas-end" tabindex="-1" id="offcanvasNavbar"
aria-labelledby="offcanvasNavbarLabel">
            <div class="offcanvas-header">
              <h5 class="offcanvas-title" id="offcanvasNavbarLabel">Converse</h5>
              <button type="button" class="btn-close" data-bs-dismiss="offcanvas"
aria-label="Close"></button>
            </div>
            <div class="offcanvas-body">
              <ul class="navbar-nav justify-content-end flex-grow-1 pe-3">
                <li class="nav-item">
                  <a class="nav-link" href="///127.0.0.1:8000/{% url 'about'
%}">About Us</a>
                </li>
                <li class="nav-item">
                  <a class="nav-link" aria-current="page"
href="///127.0.0.1:8000">Chatroom</a>
                </li>
                <li class="nav-item">
                  <a class="nav-link" {% if 'user' in request.session %}
                  href="live_chat"
                {% else %}
                  href="live_chat/login"
                {% endif %}>Direct Chat</a>
                </li>
              </ul>
              <form class="d-flex mt-3" role="search">
                <a href="{% url '/logout' %}" id="inout"></a>
              </form>
            </div>
          </div>
        </div>
      </nav>
    {% block content %}
    {% endblock content %}
    <script>
      let inout=document.getElementById("inout")
      {% if 'user' in request.session %}
        inout.innerText="Logout"
      {% else %}
        inout.innerText="Login"
      {% endif %}
    </script>
</body>
{% block message_ajax %}
{% endblock message_ajax %}
</html>
```

index. html

```
{% extends 'chat/base.html' %}

{% block content %}
<style>
  body{
    background-color : #2c2c2c;
    color:#cbb300;
  }
  .btn-primary{
    background-color:#cbb300;
    border : 1px solid white;
  }
</style>
<div class="container">
    <form method="POST" action="checkroom">
        {% csrf_token %}
        <div class="mb-3">
          <label for="exampleInputEmail1" class="form-label">Room Name</label>
          <input type="text" class="form-control" id="exampleInputEmail1"
name="room_name">
          {% comment %} <div id="emailHelp" class="form-text">We'll never share
your email with anyone else.</div> {% endcomment %}
        </div>
        <div class="mb-3">
          <label for="exampleInputPassword1" class="form-label">Your Name</label>
          <input type="text" class="form-control" name="user_name"
id="exampleInputPassword1">
        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
      </form>
</div>
{% endblock content %}
```

room. html

```
{% extends 'chat/base.html' %}

{% block content %}
<style>
  body{
    background-color:#2c2c2c;
    color:#fff;
  }
  .roomnaam{
    display:block;
    width:40px;
    margin:40px auto;
    color:#cbb300;
  }
  .form-control{
    min-height:60px;
  }
  .btn-primary{
    background-color: #cbb300;
    border: 1px solid #fff;
    display: block;
    margin: 5px auto;
  }
</style>
<h2 class="roomnaam"> {{ room }}</h2>

<script>
    $(document).ready(function(){

    setInterval(function(){
        $.ajax({
            type: 'GET',
            url : "/getMessages/{{room}}/",
            success: function(response){
                $("#display").empty();
                for (var key in response.messages)
                {
                    var temp="<div class='alert alert-secondary'
role='alert'><strong>"+response.messages[key].user+"</strong></b><p>"+response.me
ssages[key].value+"</p><span class='time-
left'>"+response.messages[key].date+"</span></div>";
                    $("#display").append(temp);
                }
            },
            error: function(response){
                alert('An error occured')
            }
        });
    },2000);
    })
    </script>


<form id="post-form">
    {% csrf_token %}
    <div class="mb-3">
           <input type="hidden" class="form-control" name="username"
id="username" value={{username}} >
      <input type="hidden" class="form-control" name="room_id" id="room_id"
value={{room_details.id}}>
      <input type="text" class="form-control" name="message" id="message">
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
  </form>
  <div id="display">

</div>

{% endblock content %}

{% block message_ajax %}
<script type="text/javascript">
    $(document).on('submit','#post-form',function(e){
      e.preventDefault();

      $.ajax({
        type:'POST',
        url:'/send',
        data:{
            username:$('#username').val(),
            room_id:$('#room_id').val(),
            message:$('#message').val(),
          csrfmiddlewaretoken:$('input[name=csrfmiddlewaretoken]').val(),
        },
        success: function(data){
           alert(data)
        }
      });
      document.getElementById('message').value = ''
    });
  </script>

{% endblock message_ajax %}
```

➔ Live_chat

About.html

```
{% extends 'chat/base.html' %}

{% block content %}
<style>
    .container{
        background-color:#eeedf2;
        border-radius:50px;
    }
    .br30{
        border-radius:30px;
        margin-left: 100px;
        margin-top: 30px;
    }
</style>
<div class="container">

    <img class="br30" src="http://unblast.com/wp-content/uploads/2020/10/Live-Chat-Vector-Illustration.jpg" width="500" height="500">
    <img class="br30" src="https://img.freepik.com/free-vector/online-consulting-chat-research-recommendation-sales-strategy-recomendation-troubleshooting-help-clients-with-business-problems-flat-vector-illustration_613284-1895.jpg?w=2000" width="500" height="400">


</br></br><h3><strong>Privacy Policy for Converse</h3></strong></br>
At Converse, accessible from https://www.Converse.in, one of our main priorities is the privacy of our visitors. This Privacy Policy document contains types of information that is collected and recorded by Converse and how we use it.</br>

If you have additional questions or require more information about our Privacy Policy, do not hesitate to contact us.</br>
```

```
This Privacy Policy applies only to our online activities and is valid for
visitors to our website with regards to the information that they shared and/or
collect in Converse. This policy is not applicable to any information collected
offline or via channels other than this website.</br>

</br></br><h3><strong>Consent</h3></strong></br>
By using our website, you hereby consent to our Privacy Policy and agree to its
terms. For our Terms and Conditions, please visit the Terms & Conditions
Generator.</br>

</br></br><h3><strong>Information we collect</h3></strong></br>
The personal information that you are asked to provide, and the reasons why you
are asked to provide it, will be made clear to you at the point we ask you to
provide your personal information.</br>

If you contact us directly, we may receive additional information about you such
as your name, email address, phone number, the contents of the message and/or
attachments you may send us, and any other information you may choose to
provide.</br>

When you register for an Account, we may ask for your contact information,
including items such as name, company name, address, email address, and telephone
number.</br>

</br></br><h3><strong>How we use your information</h3></strong></br>
We use the information we collect in various ways, including to:</br>

Provide, operate, and maintain our webste</br>
Improve, personalize, and expand our webste</br>
Understand and analyze how you use our webste</br>
Develop new products, services, features, and functionality</br>
Communicate with you, either directly or through one of our partners, including
for customer service, to provide you with updates and other information relating
to the webste, and for marketing and promotional purposes</br>
Send you emails</br>
Find and prevent fraud</br>
Log Files</br>
Converse follows a standard procedure of using log files. These files log
visitors when they visit websites. All hosting companies do this and a part of
hosting services' analytics. The information collected by log files include
internet protocol (IP) addresses, browser type, Internet Service Provider (ISP),
date and time stamp, referring/exit pages, and possibly the number of clicks.
These are not linked to any information that is personally identifiable. The
purpose of the information is for analyzing trends, administering the site,
tracking users' movement on the website, and gathering demographic
information.</br>

<img class="br30" src="http://unblast.com/wp-content/uploads/2020/05/Group-Chat-
Illustration.jpg" width="500" height="400">

</br></br><h3><strong>Cookies and Web Beacons</h3></strong></br>
Like any other website, Converse uses 'cookies'. These cookies are used to store
information including visitors' preferences, and the pages on the website that
the visitor accessed or visited. The information is used to optimize the users'
experience by customizing our web page content based on visitors' browser type
and/or other information.

For more general information on cookies, please read "What Are Cookies" from
Cookie Consent.</br>

</br></br><h3><strong>Advertising Partners Privacy Policies</h3></strong></br>
You may consult this list to find the Privacy Policy for each of the advertising
partners of Converse.</br>

Third-party ad servers or ad networks uses technologies like cookies, JavaScript,
or Web Beacons that are used in their respective advertisements and links that
appear on Converse, which are sent directly to users' browser. They automatically
receive your IP address when this occurs. These technologies are used to measure
the effectiveness of their advertising campaigns and/or to personalize the
advertising content that you see on websites that you visit.

Note that Converse has no access to or control over these cookies that are used
by third-party advertisers.</br>

</br></br><h3><strong>Third Party Privacy Policies</h3></strong></br>
Converse's Privacy Policy does not apply to other advertisers or websites. Thus,
we are advising you to consult the respective Privacy Policies of these third-
party ad servers for more detailed information. It may include their practices
and instructions about how to opt-out of certain options.</br>

You can choose to disable cookies through your individual browser options. To
know more detailed information about cookie management with specific web
browsers, it can be found at the browsers' respective websites.</br>

</br></br><h3><strong>CCPA Privacy Rights (Do Not Sell My Personal
Information)</h3></strong></br>
Under the CCPA, among other rights, California consumers have the right to:</br>
```

```html
Request that a business that collects a consumer's personal data disclose the
categories and specific pieces of personal data that a business has collected
about consumers.</br>

Request that a business delete any personal data about the consumer that a
business has collected.</br>

Request that a business that sells a consumer's personal data, not sell the
consumer's personal data.</br>

If you make a request, we have one month to respond to you. If you would like to
exercise any of these rights, please contact us.</br>

<img class="br30" src="https://img.freepik.com/free-vector/hand-holding-phone-
with-conversation-girl-chat-bot-mobile-app-talking-robot-online-flat-vector-
illustration-technology-assistance-concept-banner-website-design-landing-
page_74855-24649.jpg?w=2000
" width="500" height="400">


</br></br><h3><strong>GDPR Data Protection Rights</h3></strong></br>
We would like to make sure you are fully aware of all of your data protection
rights. Every user is entitled to the following:</br>

The right to access – You have the right to request copies of your personal data.
We may charge you a small fee for this service.</br>

The right to rectification – You have the right to request that we correct any
information you believe is inaccurate. You also have the right to request that we
complete the information you believe is incomplete.</br>

The right to erasure – You have the right to request that we erase your personal
data, under certain conditions.</br>

The right to restrict processing – You have the right to request that we restrict
the processing of your personal data, under certain conditions.</br>

The right to object to processing – You have the right to object to our
processing of your personal data, under certain conditions.</br>

The right to data portability – You have the right to request that we transfer
the data that we have collected to another organization, or directly to you,
under certain conditions.</br>
```

```
If you make a request, we have one month to respond to you. If you would like to
exercise any of these rights, please contact us.</br>

</br></br><h3><strong>Children's Information</h3></strong></br>
Another part of our priority is adding protection for children while using the
internet. We encourage parents and guardians to observe, participate in, and/or
monitor and guide their online activity.</br>

Converse does not knowingly collect any Personal Identifiable Information from
children under the age of 13. If you think that your child provided this kind of
information on our website, we strongly encourage you to contact us immediately
and we will do our best efforts to promptly remove such information from our
records.</br>
</br>
</div>
{% endblock content %}
```

Detail.html

```
{% extends 'chat/base.html' %}

{% block content %}
<script>
  .bscb{
    box-sizing : content-box;
    display : flex;
  }
  .uname{

    background-color:red;
  }

</script>
<div class="container">

<h2 class="badge text-bg-dark" style="position:sticky;top:60px;width:100%;font-
size:2rem;">{{friend.profile.name}}</h2>
    <div id="chat-body" class="container" style="position:inherit ;
top:500px;background-color : silver;display : flex; flex-direction:column;min-
height:570px;border-radius:30px;padding-top:30px;">

          {% for chat in chats %}
          {% if chat.msg_sender == profile and chat.msg_reciever == user %}
            <div class="d-flex flex-row mb-3 bscb" style="">
              <span class="badge text-bg-primary" ><p
class="h5">{{chat}}</p></span>
            </div>
          {% elif chat.msg_sender == user and chat.msg_reciever == profile%}
            <div class="d-flex flex-row-reverse mb-3 bscb">
              <span class="badge text-bg-success"><p
class="h5">{{chat}}</p></span>
            </div>
          {% endif %}
          {% endfor %}

    </div>
    <form method="POST" id="myform">
        <div class="mb-3 d-flex">
          {% comment %} <input type="text" class="form-control"
id="exampleInputEmail1" style="width:1000px;" > {% endcomment %}
          {% csrf_token %}
          {{form.body}}

        <button type="submit" class="btn btn-primary">Submit</button>
      </div>
    </form>
</div>

<script>
  window.scrollTo(0, document.body.scrollHeight);
  function getCookie(name) {
    let cookieValue = null;
    if (document.cookie && document.cookie !== '') {
        const cookies = document.cookie.split(';');
        for (let i = 0; i < cookies.length; i++) {
            const cookie = cookies[i].trim();
            // Does this cookie string begin with the name we want?
            if (cookie.substring(0, name.length + 1) === (name + '=')) {
                cookieValue = decodeURIComponent(cookie.substring(name.length +
1));
                break;
            }
        }
    }
    return cookieValue;
  }
```

```
const csrftoken = getCookie('csrftoken');

let form=document.getElementById("myform");
form.addEventListener("submit" , sendChat);

function sendChat(e){

  e.preventDefault()
  let chatMessage= document.getElementById("iid3").value
  console.log(chatMessage)

  const data = { msg: chatMessage };
  let url= "{% url 'sent_msg' pk=friend.profile.id %}"

fetch(url, {
  method: 'POST', // or 'PUT'
  headers: {
    'Content-Type': 'application/json',
    'X-CSRFToken': csrftoken,
  },
  body: JSON.stringify(data),
})
  .then((response) => response.json())
  .then((data) => {
    console.log('Success:', data);
    let chat_body= document.getElementById("chat-body")
    let chatMessageBox=document.createElement("div")
    chatMessageBox.classList.add("d-flex")
    chatMessageBox.classList.add("bscb")
    chatMessageBox.classList.add("flex-row-reverse")
    chatMessageBox.classList.add("mb-3")
    //<span class="badge text-bg-success"><p class="h5">{{chat}}</p></span>
    let sent_chat=document.createElement("span")
    sent_chat.classList.add("badge")
    sent_chat.classList.add("text-bg-success")
    let tempvar=document.createElement("p")
    tempvar.classList.add("h5")
    tempvar.innerText=data
    sent_chat.append(tempvar)
    chatMessageBox.append(sent_chat)
    chat_body.append(chatMessageBox)

    document.getElementById("iid3").value=""
    window.scrollTo(0, document.body.scrollHeight);
  })
  .catch((error) => {
    console.error('Error:', error);
  });
}
setInterval( recieveMessages , 3000);

let counter={{num}}
function recieveMessages(){


  let url= "{% url 'rec_msg' pk=friend.profile.id %}"

fetch(url)
  .then((response) => response.json())
  .then((data) => {
    console.log('Success:', data);

    if (data.length == 0){}

    else{
      let lastMsg = data[data.length-1]
      if (counter==data.length){console.log("There are no new messages")}
      else{
        let chat_body= document.getElementById("chat-body")
        let chatMessageBox=document.createElement("div")
        chatMessageBox.classList.add("d-flex")
        chatMessageBox.classList.add("bscb")
        chatMessageBox.classList.add("flex-row")
        chatMessageBox.classList.add("mb-3")
        //<span class="badge text-bg-success"><p class="h5">{{chat}}</p></span>
        let sent_chat=document.createElement("span")
        sent_chat.classList.add("badge")
        sent_chat.classList.add("text-bg-primary")
        let tempvar=document.createElement("p")
        tempvar.classList.add("h5")
        tempvar.innerText=lastMsg
        sent_chat.append(tempvar)
        chatMessageBox.append(sent_chat)
        chat_body.append(chatMessageBox)
        document.getElementById("iid3").value=""
      }
    }

    counter=data.length
  })
```

```
    .catch((error) => {
      console.error('Error:', error);
    });
  }
</script>
{% endblock content %}
```

Forgot.html

```
{% extends 'chat/base.html' %}
{% block content %}
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="author" content="Yinka Enoch Adedokun">
  <meta name="description" content="Simple Forgot Password Page Using HTML and
CSS">
  <meta name="keywords" content="forgot password page, basic html and css">
  <title>Forgot Password Page - HTML + CSS</title>
</head>
<style>
  * {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "segoe ui", verdana, helvetica, arial, sans-serif;
  font-size: 16px;
  transition: all 500ms ease; }

body {
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-rendering: optimizeLegibility;
  -moz-font-feature-settings: "liga" on;
  background-color:#2c2c2c;
}
.row {
  background-color: #2c2c2c;
  color: #cbb300;
  text-align: center;
  padding: 2em 2em 0.5em;
  border:2px solid;
```

```
width: 90%;
margin: 2em auto;
border-radius: 5px; }
.row h1 {
  font-size: 2.5em; }
.row .form-group {
  margin: 0.5em 0; }
  .row .form-group label {
    display: block;
    color: #cbb300;
    text-align: left;
    font-weight: 600; }
  .row .form-group input, .row .form-group a {
    display: block;
    padding: 0.5em 0;
    width: 100%;
    margin-top: 1em;
    margin-bottom: 0.5em;
    background-color: inherit;
    border: none;
    border-bottom: 1px solid #555;
    color: #eee; }
    .row .form-group input:focus, .row .form-group a:focus {
      background-color: #cbb300;
      color: #000;
      border: none;
      padding: 1em 0.5em; animation: pulse 1s infinite ease;}
  .row .form-group a {
    border: 1px solid #cbb300;
    text-decoration:none;
    border-radius: 5px;
    outline: none;
    -moz-user-select: none;
    user-select: none;
    color: #fff;
    font-weight: 800;
    cursor: pointer;
    margin-top: 2em;
    padding: 1em; }
    .row .form-group a:hover, .row .form-group a:focus {
      background-color: #cbb300; }

.row .footer h5 {
  margin-top: 1em; }
.row .footer p {
      margin-top: 2em; }
      .row .footer p .symbols {
        color: #444; }
    .row .footer a {
      color: inherit;
      text-decoration: none; }

  .information-text {
    color: #ddd; }

  @media screen and (max-width: 320px) {
    .row {
      padding-left: 1em;
      padding-right: 1em; }
      .row h1 {
        font-size: 1.5em !important; } }
  @media screen and (min-width: 900px) {
    .row {
      width: 50%; } }

  </style>
  <body>
      <form method="POST">
          {% csrf_token %}
    <div class="row">
      <h1>Forgot Password</h1>
      <h6 class="information-text">Enter your registered email to reset your
  password.</h6>
      <div class="form-group">
              <div id="fg">
        <input type="email" name="user_email" id="user_email">
        <p><label id='lbl' for="username">Email</label></p>
              </div>
        <button id='reset_btn' >Reset Password</button>
      </div>
      <div class="footer">
        <h5>New here? <a href="///127.0.0.1:8000/{% url '/login' %}">Sign
  Up.</a></h5>
        <h5>Already have an account? <a href="///127.0.0.1:8000/{% url '/login'
  %}">Sign In.</a></h5>

      </div>
    </div>
  </body>
  <script>
```

```
    function getCookie(name) {
      let cookieValue = null;
      if (document.cookie && document.cookie !== '') {
          const cookies = document.cookie.split(';');
          for (let i = 0; i < cookies.length; i++) {
              const cookie = cookies[i].trim();
              // Does this cookie string begin with the name we want?
              if (cookie.substring(0, name.length + 1) === (name + '=')) {
                  cookieValue = decodeURIComponent(cookie.substring(name.length +
  1));
                  break;
              }
          }
      }
      return cookieValue;
    }
    const csrftoken = getCookie('csrftoken');
      let reset_btn=document.getElementById('reset_btn')
      let fg=document.getElementById('fg')
      let lbl=document.getElementById('lbl')
      reset_btn.onclick = function(e){
          e.preventDefault()
          let em=document.getElementById("user_email").value
          let ott=document.createElement("input")
          let lbl2=document.createElement("label")
          ott.id = 'ootp'
          lbl2.innerText="Enter OTP"
          console.log(em)
          fg.append(ott)
          fg.append(lbl2)
          const data = { em: em };
      let url= "{% url 'ver_em' %}"

    fetch(url, {
      method: 'POST', // or 'PUT'
      headers: {
        'Content-Type': 'application/json',
        'X-CSRFToken': csrftoken,
      },
      body: JSON.stringify(data),
    })
      .then((response) => response.json())
      .then((data) => {
        console.log('Success:', data);
        let new_btn=document.createElement("button")
        new_btn.innerText="Verify OTP"
        fg.append(new_btn)
        reset_btn.style.display="none"
        new_btn.onclick= function(e){
          e.preventDefault()
          let ootp=document.getElementById("ootp").value
          if(ootp == data){
              window.location.replace("reset");
          }
        }
      })
      .catch((error) => {
        console.error('Error:', error);
      });
      };
  </script>
  {% endblock content %}
```

Index.html

```
{% extends 'chat/base.html' %}
{% block content %}
<style>
  body{
    background-color : #2c2c2c;
  }
  .naam{
    margin: 2px;
    border: 5px solid #cbb300;
    min-height : 60px;
  }
</style>
<div class="container">
    {% for friend in friends %}
    <a href="{% url 'detail' pk=friend.profile.id %}" style="text-
decoration:none;">
<ul class="list-group list-group">

    <li class="naam list-group-item d-flex justify-content-between align-items-
start">
      <div class="ms-2 me-auto">
        <div class="fw-bold">{{friend.profile.name}}</div>
      </div>
      <span id="notify" class="notify badge bg-primary rounded-pill"></span>
    </li>
  </ul>
</a>
  {% endfor %}
</div>
<script>
  setInterval(getNotification , 1000)
  function getNotification(){
    let url = "{% url 'notification' %}"
    fetch(url)
    .then((response) => response.json())
    .then(data => {
      console.log(data)
      let chatNotificationBtn = document.getElementsByClassName("notify")
      for (let i=0; i<data.length; i++){
        chatNotificationBtn[i].innerText=data[i]
      }
    })
    .catch((error) => {
      console.error('Error:', error);
    });
  }
</script>

{% endblock content %}
```

Login.html

```
{% extends 'chat/base.html' %}
{% block content %}

<style>

    @import
url('https://fonts.googleapis.com/css?family=Poppins:400,500,600,700&display=swap
');
    *{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
```

```
font-family: 'Poppins', sans-serif;
}
html,body{
display: grid;
height: 100%;
width: 100%;
place-items: center;
background: -webkit-linear-gradient(left, #2e2e2e,#000000,#272727
, #434343);
}
::selection{
background: #d5be0f;
color: #fff;
}
.wrapper{
overflow: hidden;
max-width: 390px;
background: #fff;
padding: 30px;
border-radius: 15px;
box-shadow: 0px 15px 20px rgba(0,0,0,0.1);
}
.wrapper .title-text{
display: flex;
width: 200%;
}
.wrapper .title{
width: 50%;
font-size: 35px;
font-weight: 600;
text-align: center;
transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);
}
.wrapper .slide-controls{
position: relative;
display: flex;
height: 50px;
width: 100%;
overflow: hidden;
margin: 30px 0 10px 0;
justify-content: space-between;
border: 1px solid lightgrey;
border-radius: 15px;
}
.slide-controls .slide{
```

```
height: 100%;
width: 100%;
color: rgb(30, 28, 18);
font-size: 18px;
font-weight: 500;
text-align: center;
line-height: 48px;
cursor: pointer;
z-index: 1;
transition: all 0.6s ease;
}
.slide-controls label.signup{
color: #000;
}
.slide-controls .slider-tab{
position: absolute;
height: 100%;
width: 50%;
left: 0;
z-index: 0;
border-radius: 15px;
background: -webkit-linear-gradient(left,#dac207,#d9c000,#fff186
, #fff7bc);
transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);
}
input[type="radio"]{
display: none;
}
#signup:checked ~ .slider-tab{
left: 50%;
}
#signup:checked ~ label.signup{
color: rgb(30, 28, 18);
cursor: default;
user-select: none;
}
#signup:checked ~ label.login{
color: #000;
}
#login:checked ~ label.signup{
color: #000;
}
#login:checked ~ label.login{
cursor: default;
user-select: none;
```

```
}
.wrapper .form-container{
width: 100%;
overflow: hidden;
}
.form-container .form-inner{
display: flex;
width: 200%;
}
.form-container .form-inner form{
width: 50%;
transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);
}
.form-inner form .field{
height: 50px;
width: 100%;
margin-top: 20px;
}
.form-inner form .field input{
height: 100%;
width: 100%;
outline: none;
padding-left: 15px;
border-radius: 15px;
border: 1px solid lightgrey;
border-bottom-width: 2px;
font-size: 17px;
transition: all 0.3s ease;
}
.form-inner form .field input:focus{
border-color: #d5be0f;
/* box-shadow: inset 0 0 3px #fb6aae; */
}
.form-inner form .field input::placeholder{
color: #999;
transition: all 0.3s ease;
}
form .field input:focus::placeholder{
color: #d5be0f;
}
.form-inner form .pass-link{
margin-top: 5px;
}
.form-inner form .signup-link{
text-align: center;
```

```
margin-top: 30px;
}
.form-inner form .pass-link a,
.form-inner form .signup-link a{
color: #d5be0f;
text-decoration: none;
}
.form-inner form .pass-link a:hover,
.form-inner form .signup-link a:hover{
text-decoration: underline;
}
form .btn{
height: 50px;
width: 100%;
border-radius: 15px;
position: relative;
overflow: hidden;
}
form .btn .btn-layer{
height: 100%;
width: 300%;
position: absolute;
left: -100%;
background: -webkit-linear-gradient(right,#f8e342,#ffe93f,#fff7bb
, #d5be0f);
border-radius: 15px;
transition: all 0.4s ease;;
}
form .btn:hover .btn-layer{
left: 0;
}
form .btn input[type="submit"]{
height: 100%;
width: 100%;
z-index: 1;
position: relative;
background: none;
border: none;
color: rgb(0, 0, 0);
padding-left: 0;
border-radius: 15px;
font-size: 20px;
font-weight: 500;
cursor: pointer;
}
```

```
  .error{
    color:red;
  }

</style>

<div class="wrapper">
    <div class="title-text">
      <div class="title login">Login Form</div>
      <div class="title signup">Signup Form</div>
    </div>
    <div class="form-container">
      <div class="slide-controls">
        <input type="radio" name="slide" id="login" checked>
        <input type="radio" name="slide" id="signup">
        <label for="login" class="slide login">Login</label>
        <label for="signup" class="slide signup">Signup</label>
        <div class="slider-tab"></div>
      </div>
      <div class="form-inner">
        <form action="" method="POST" class="login" id="loginform">
          {% csrf_token %}
          <div class="field">
            <input type="text" placeholder="Email Address" id="logeMail"
name="logeMail" required>
          </div>
          <div class="field">
            <input type="password" placeholder="Password" id="logpassword"
name="logpassword" required>
          </div>
          <div class="pass-link"><a href="///127.0.0.1:8000/{% url '/forgot'
%}">Forgot password?</a></div>
          <div class="field btn">
            <div class="btn-layer"></div>
            <input type="submit" value="Login">
          </div>
          <div class="signup-link">Not a member? <a href="">Signup now</a></div>
        </form>
        <form action="" class="signup" id="signupform">
          {% csrf_token %}
          <div class="field">
            <input type="text" placeholder="User Name" id="userName"
name="userName" required>
          </div>
          <p id="nameerr" class="error"></p>
          <div class="field">
            <input type="text" placeholder="Email Address" id="eMail"
name="eMail" required>
          </div>
          <div class="field">
            <input type="password" placeholder="Password" id="password1"
name="password1" required>
          </div>
          <div class="field">
            <input type="password" placeholder="Confirm password" id="password2"
name="password2" required>
          </div>
          <p id="passerr" class="error"></p>

          <div class="field btn">
            <div class="btn-layer"></div>
            <input type="submit" value="Signup">
          </div>
        </form>
      </div>
    </div>
  </div>

  <script>
  const loginText = document.querySelector(".title-text .login");
  const loginForm = document.querySelector("form.login");
  const loginBtn = document.querySelector("label.login");
  const signupBtn = document.querySelector("label.signup");
  const signupLink = document.querySelector("form .signup-link a");
  signupBtn.onclick = (()=>{
    loginForm.style.marginLeft = "-50%";
    loginText.style.marginLeft = "-50%";
  });
  loginBtn.onclick = (()=>{
    loginForm.style.marginLeft = "0%";
    loginText.style.marginLeft = "0%";
  });
  signupLink.onclick = (()=>{
    signupBtn.click();
    return false;
  });
  function getCookie(name) {
    let cookieValue = null;
    if (document.cookie && document.cookie !== '') {
        const cookies = document.cookie.split(';');
        for (let i = 0; i < cookies.length; i++) {
            const cookie = cookies[i].trim();
            // Does this cookie string begin with the name we want?
            if (cookie.substring(0, name.length + 1) === (name + '=')) {
                cookieValue = decodeURIComponent(cookie.substring(name.length +
1));
                break;
            }
        }
    }
    return cookieValue;
  }
  const csrftoken = getCookie('csrftoken');

  let form=document.getElementById("signupform");
  let userName=document.getElementById("userName")
  let eMail=document.getElementById("eMail")
  let password1=document.getElementById("password1")
  let password2=document.getElementById("password2")
  form.addEventListener("submit" , signupform);
  function signupform(e){
    e.preventDefault()
    const data={ userName : userName.value , eMail : eMail.value , password1 :
password1.value , password2 : password2.value }
    console.log(data)
    let url= "{% url 'handle_signup' %}"
    fetch(url, {
      method: 'POST', // or 'PUT'
      headers: {
        'Content-Type': 'application/json',
        'X-CSRFToken': csrftoken,
      },
      body: JSON.stringify(data),
    })
      .then((response) => response.json())
      .then((data) => {
        console.log('Success:', data);
        if(data=="Name Error"){
          document.getElementById("nameerr").innerText = "This Username already
exists";
        }
        else if(data=="password Error"){
          document.getElementById("passerr").innerText = "Passwords does not
match";
        }
        else{
          document.getElementById("login").checked = true;
          loginForm.style.marginLeft = "0%";
          loginText.style.marginLeft = "0%";
          document.getElementById("userName").value=""
          document.getElementById("eMail").value=""
          document.getElementById("password1").value=""
          document.getElementById("password2").value=""
        }
      })
      .catch((error) => {
        console.error('Error:', error);
      });
    }
</script>
{% endblock content %}
```

Reset.html

```
{% extends 'chat/base.html' %}
{% block content %}

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="author" content="Yinka Enoch Adedokun">
  <meta name="description" content="Simple Forgot Password Page Using HTML and
CSS">
  <meta name="keywords" content="forgot password page, basic html and css">
  <title>Forgot Password Page - HTML + CSS</title>
</head>

<style>
  * {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "segoe ui", verdana, helvetica, arial, sans-serif;
  font-size: 16px;
  transition: all 500ms ease; }

body {
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-rendering: optimizeLegibility;
  -moz-font-feature-settings: "liga" on;
  background-color:#2c2c2c;
}

.row {
  background-color: #2c2c2c;
  color: #cbb300;
  text-align: center;
  padding: 2em 2em 0.5em;
  border:2px solid;
  width: 90%;
  margin: 2em auto;
  border-radius: 5px; }
  .row h1 {
    font-size: 2.5em; }
  .row .form-group {
    margin: 0.5em 0; }
    .row .form-group label {
      display: block;
      color: #cbb300;
      text-align: left;
      font-weight: 600; }
    .row .form-group input, .row .form-group a {
      display: block;
      padding: 0.5em 0;
      width: 100%;
      margin-top: 1em;
      margin-bottom: 0.5em;
      background-color: inherit;
      border: none;
      border-bottom: 1px solid #555;
      color: #eee; }
      .row .form-group input:focus, .row .form-group a:focus {
        background-color: #cbb300;
        color: #000;
        border: none;
        padding: 1em 0.5em; animation: pulse 1s infinite ease;}
    .row .form-group a {
      border: 1px solid #cbb300;
      text-decoration:none;
      border-radius: 5px;
      outline: none;
```

```
            -moz-user-select: none;
            user-select: none;
            color: #fff;
            font-weight: 800;
            cursor: pointer;
            margin-top: 2em;
            padding: 1em; }
            .row .form-group a:hover, .row .form-group a:focus {
              background-color: #cbb300; }

        .row .footer h5 {
          margin-top: 1em; }
        .row .footer p {
          margin-top: 2em; }
          .row .footer p .symbols {
            color: #444; }
        .row .footer a {
          color: inherit;
          text-decoration: none; }

      .information-text {
        color: #ddd; }

      @media screen and (max-width: 320px) {
        .row {
          padding-left: 1em;
          padding-right: 1em; }
          .row h1 {
            font-size: 1.5em !important; } }
      @media screen and (min-width: 900px) {
        .row {
          width: 50%; } }

      </style>
      <body>
          <form method="POST">
              {% csrf_token %}
        <div class="row">
          <h1>Reset Password</h1>
          <h6 class="information-text">Enter new password.</h6>
              <form>
          <div class="form-group">
                  <div id="fg">
                      <input type="password" name="pass1" id="pass1">
                      <p><label id='lbl' for="pass1">Password</label></p>
```

```
                <input type="password" name="pass2" id="pass2">
          <p><label id='lbl' for="pass2">Re-Enter Password</label></p>
              </div>
          <button id='reset_btn' >Reset Password</button>
        </div>
        </form>
        <div class="footer">
          <h5>New here? <a href="///127.0.0.1:8000/{% url '/login' %}">Sign
Up.</a></h5>
          <h5>Already have an account? <a href="///127.0.0.1:8000/{% url '/login'
%}">Sign In.</a></h5>
        </div>
      </div>
</body>
<script>

    function getCookie(name) {
        let cookieValue = null;
        if (document.cookie && document.cookie !== '') {
            const cookies = document.cookie.split(';');
            for (let i = 0; i < cookies.length; i++) {
                const cookie = cookies[i].trim();
                // Does this cookie string begin with the name we want?
                if (cookie.substring(0, name.length + 1) === (name + '=')) {
                    cookieValue = decodeURIComponent(cookie.substring(name.length
+ 1));

                    break;
                }
            }
        }
        return cookieValue;
      }
      const csrftoken = getCookie('csrftoken');

    let reset_btn=document.getElementById("reset_btn")
    reset_btn.onclick= function(e){
        e.preventDefault()
        let p1 = document.getElementById("pass1")
    let p2 = document.getElementById("pass2")
    if( p1.value == p2.value ){

        const data = { pd: p1.value };
    let url= "{% url 'change_pass' %}"

  fetch(url, {
    method: 'POST', // or 'PUT'
    headers: {
      'Content-Type': 'application/json',
      'X-CSRFToken': csrftoken,
    },
    body: JSON.stringify(data),
  })
    .then((response) => response.json())
    .then((data) => {
      console.log('Success:', data);

        window.location.replace("login");
    })
    .catch((error) => {
      console.error('Error:', error);
    });
    };
    //    window.location.replace("change_pass");
    }
</script>
{% endblock content %}
```

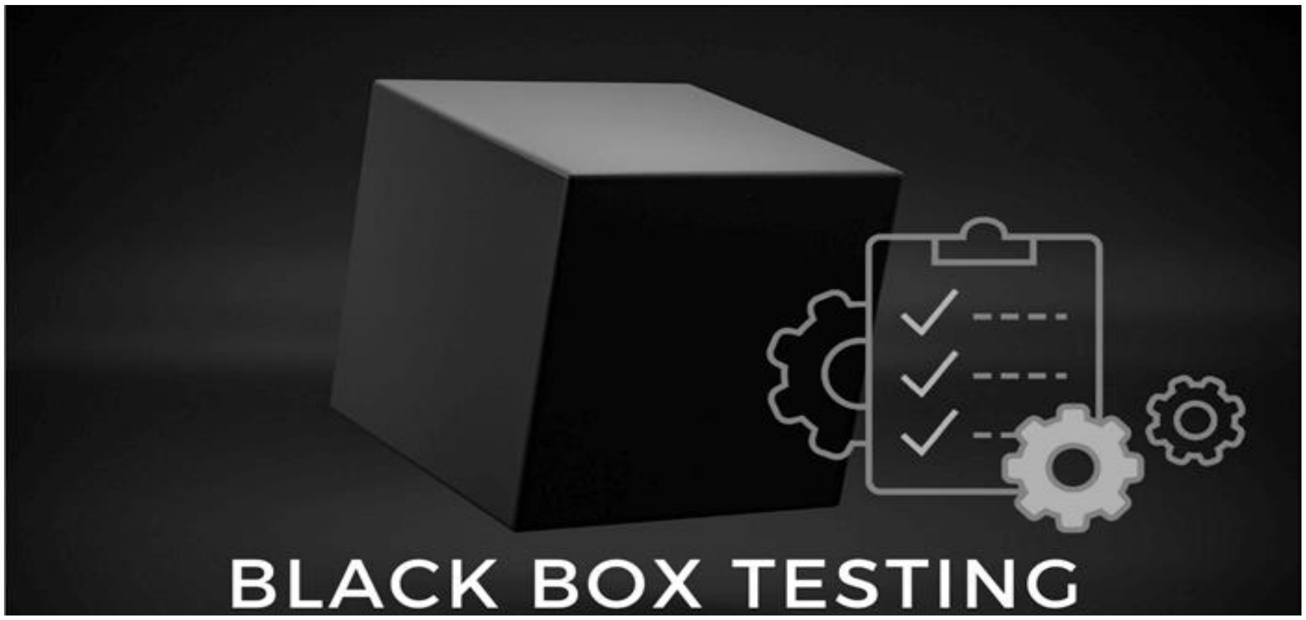
☯ Testing ☯

Software Testing involves the evaluation of the functionality of a software application to find out the software bugs. It checks if the developed website met the specified requirements and identifies the website defect, if any, to produce a quality product. Software testing

- Works as per the requirements.
- Meets the business and technical requirements, guiding its design and development.
- Can be implemented using similar characteristics.

Black box Testing
White box Testing
Gray box Testing
+
=
Testing Techniques To Stay Bug-Free

➢Black Box Testing:



Black Box is a software testing type that has nothing to do with the internal structure or workings of the codes. The software tester doesn't need to know the system architecture or source code. Black box testing is also known as closed-box testing, data driven testing or functional testing. A functionality of an application can be tested by interacting with its user interface. A tester provides inputs, records outputs and compares it with the standard output he expects. Unexpected results and deviations are noted and brought to the notice of developers and designers.

➢White Box Testing:

White box testing is an approach that allows testers to inspect and verify the inner workings of a software system—its code, infrastructure, and integrations with external systems. White box testing is an essential part of automated build processes in a modern Continuous Integration/Continuous Delivery (CI/CD) development pipeline.

☯ Future Enhancement ☯

➢ An android app launch

An android app can be launched for this chatting web app. Users can use the app to chat as well as if their mobile is not accessible to them then they can log in through any device with an internet connection and a browser.

Mobile apps are easy to operate and load faster. Hence they are highly preferred in the social media industry. Users do not have to go through a tedious process of opening up the browser, typing the URL, on top of that, waiting for the page to load. It is much easier to click on the app icon. On top of that, what if the website is not mobile responsive? Opening up such a website in the mobile browser can be very troublesome. Hence, it is always straightforward to navigate using mobile apps.

➢ Unsending message facility

Sometimes users send messages by mistake , or sometimes they regret sending it. In that case the users can easily unsend an message with just on click.

➢ Broadcasting

Broadcasting a particular message to a group of people can save time and also reduce a lot of load from the user.

➢ Setting Timer

Setting a timer to send a particular message to a specific person can help users to work efficiently.

You can set a timer and the message will be sent exactly at that time to a particular user.

☯ Bibliography ☯

I have used Python,Django-Framework ,Css, Html, Mysqlite3 Database(Backend) and javascript to develop this website and specifically for learning Python – Django Framework or webappdevlopment.

I used the following material and website :

- https://www.djangoproject.com
- https://www.w3schools.com/django
- https://www.youtube.com
- https://realpython.com/tutorials/django/
- https://www.geeksforgeeks.org/django-tutorial/