G-Blog

An Internship Report submitted in partial fulfillment of the requirements for the award

of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

SDK Prasad,121810301006;S.Sai Sravya, 121810301031;N.Dheeraj Kumar,121810301033;B.Karthikey,121810301043

Under the esteemed guidance of

S. N. V. Jitendra M

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GITAM

(Deemed to be University)

VISAKHAPATNAM

Nov,2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING GITAM INSTITUTE OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

I/We, hereby declare that the project report entitled "**G-BLOG**" is an original work done in the Department of Computer Science and Engineering, GITAM Institute of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date:6/11/2021

Registration No(s).	Name(s)	Signature(s)
121810301006	**SDK Prasad**	
121810301031	**S.Sai Sravya**	
121810301033	**N.Dheeraj Kumar**	
121810301043	**B.Karthikey**	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled "G-CONNECT" is a bonafide record of work carried out by

SDK Prasad,121810301006;S.Sai Sravya, 121810301031;N.Dheeraj Kumar,121810301033;B.Karthikey,121810301043

students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide	**Head of the Department**
S. N. V. Jitendra M	Sireesha R
Assistant professor	**Professor**

ACKNOWLEDGEMENT

We would like to thank our project guide S.N.V. Jitendra M, Assistant Professor, Department of CSE for his stimulating guidance and profuse assistance. We shall always cherish our association with him for his guidance, encouragement and valuable suggestions throughout the progress of this work. We consider it a great privilege to work under her guidance and constant support

We also express our thanks to the project reviewers Swarna Latha Prathipati,Assistant Professor, and Srinivas Rao Gorla,Associate Professor, Department of CSE, GITAM (Deemed to be University) for their valuable suggestions and guidance for doing our project.

We consider it is a privilege to express our deepest gratitude to R.Sireesha Head of the Department, Computer Science Engineering for his valuable suggestions and constant motivation that greatly helped us to successfully complete this project and for inspiring us to learn new technologies and tools.

Finally, we deem it a great pleasure to thank one and all that helped us directly and indirectly throughout this project.

SDK PRASAD 121810301006

S.SAI SRAVYA 121810301031

N.DHEERAJ KUMAR 121810301033

B.KARTHIKEY 121810301043

TABLE OF CONTENTS

Abstract

A blog is a unique and fun method to express yourself and communicate with others. The value of multiple media as data sources has just lately been recognised by researchers. Until date, the research potential of blogs has been substantially under-utilized.

G-Blog is a website that lets students and also staff from gitam university to post information online.This information will be viewed by all the users of the website. We used python technologies like pipenv and manage.py runserver to launch the website and use it.

Introduction

The building of dynamic web applications is accomplished through web programming, often known as web development. Web applications include social networking sites such as Facebook and e-commerce sites such as Amazon.

The good news is that learning web programming isn't as difficult as it may appear.

Many feel that it is the best type of coding to learn for beginners. It's simple to set up, produces immediate effects, and there's plenty of online training.

Many people learn web coding in order to build the next Facebook or find work in the business. However, because it's so simple to get started, it's also an excellent pick if you just want a general introduction to coding. Learning how to build for the web is for you, whether you're looking for a job or just want to learn how to code.

Web Development Overview

Front-end development (also known as client-side development) and back-end development are the two types of web development (also called server-side development).

The process of generating the content, design, and interactivity that a user sees when they launch a web application is known as front-end development. This is accomplished using HTML, CSS, and JavaScript.

HTML (Hyper Text Markup Language) is a particular language for'marking up' text so that it can be turned into a web page. HTML is used to create every web page on the internet, and it will serve as the foundation for any web application. CSS stands for Cascading Style Sheets and is a programming language for defining style guidelines for web pages. CSS is in charge of the aesthetics of the web.

Finally, JavaScript is a popular programming language for adding functionality and interactivity to online sites. Back-end scripts are written in many different coding languages and frameworks, such as…

- PHP
- Ruby on Rails
- ASP.NET
- Perl
- Java
- Node.js
- Python

Why Is Website Development so Important?

As humans, we take pleasure in our exceptional adaptability and capacity to evolve with the times. As a result, when the social world evolved from one of interpersonal connections to one of internet connections, so did the economic world. There's no avoiding it: our digitally-driven existence has left us with no choice but to adapt to technology, and if you haven't done so already, now is the moment. It's no longer a question; website development is now a need for you as a business owner. Your voice must be heard, your brand must be recognised, and your objectives must be met in order for your firm to grow.Website development is the key to making such things happen. A website, like an eye, is the window to the spirit of a business, giving clients a sample of what you have to offer and tempting them to delve deeper for more.

We are simple beings that enjoy ease, and nothing beats obtaining a wealth of knowledge by simply clicking a button, which is exactly what millions of people do every day when they go online. If your product isn't located on the other side of that click, you've effectively lost access to those millions of consumers, and your business is effectively dead. In addition, creating a strong web presence as a

business and reaching millions of potential clients would elevate your product to a universally compatible entity, even if the service you provide is geographically localised to a certain location.

How is website development useful?

Website creation is a way to make people aware of the services and/or products you're giving, to explain why your products are important and even necessary for them to buy or use, and to see what distinguishes your firm from competitors. Customers will be influenced much by displaying this information with high-quality photographs and a well-thought-out presentation, thus it is critical to do so.

As much as possible, make your product relatable and appealing. Furthermore, website building allows you to:

1.Communicate effectively with your visitors. Engaging with your audience is crucial when it comes to increasing money. It is feasible to design a website that allows you to communicate with your customers and prospects, as well as relevant content for your industry or business's target audience. After that, post the content to your blog, share it on social media, and answer customer comments and feedback as soon as possible. This will show your clients that you care about their happiness and that you are attentive to their wants.

2. Boost your internet connection. Expanding your reach and drawing more people to your business will be easier with a website. Making a responsive website design for your site will allow it to be accessed by a wide range of users using a variety of devices, such as tablets and smartphones. This will boost your site's exposure as well as organic traffic.

3. Demonstrate your reliability. A website is a simple way to demonstrate a company's legitimacy, and how a person portrays his company online is critical for getting more customers or visitors. As a result, your website design should be performed as professionally as possible, because a professional presentation speaks volumes about your company. You may include your talents, credentials, experience, expertise, and more in one place with the help of website creation. These information assist you gain

your visitors' trust and confidence, as well as serve as a reference point for customers who are interested in your business, making it easier for you to generate leads.

About The Project

The foundation of this project is blogging. Blogging is the self-publication of writing, photography, and other forms of media on the internet. Blogging began as a way for people to post diary-style entries, but it has subsequently been integrated into many businesses' websites.

A blog is a sort of website that is updated with new content on a regular basis. Blog posts are short, casual writings that appear on most blogs. Text, images, videos, and other media are frequently included in these posts. A blog is essentially a platform on the Internet where you may record and share your thoughts, experiences, and interests.

Even if you don't know what a blog post is, if you spend a lot of time on the Internet, you've probably come across one. Some of the most successful blogs resemble online magazines because they're written by a team of people who are paid to update the blog with new content numerous times a day.

However, the majority of blogs are written by a single person. As a result, most blogs are very personal, reflecting the writer's unique interests and personality.

Problem Identification & Objectives

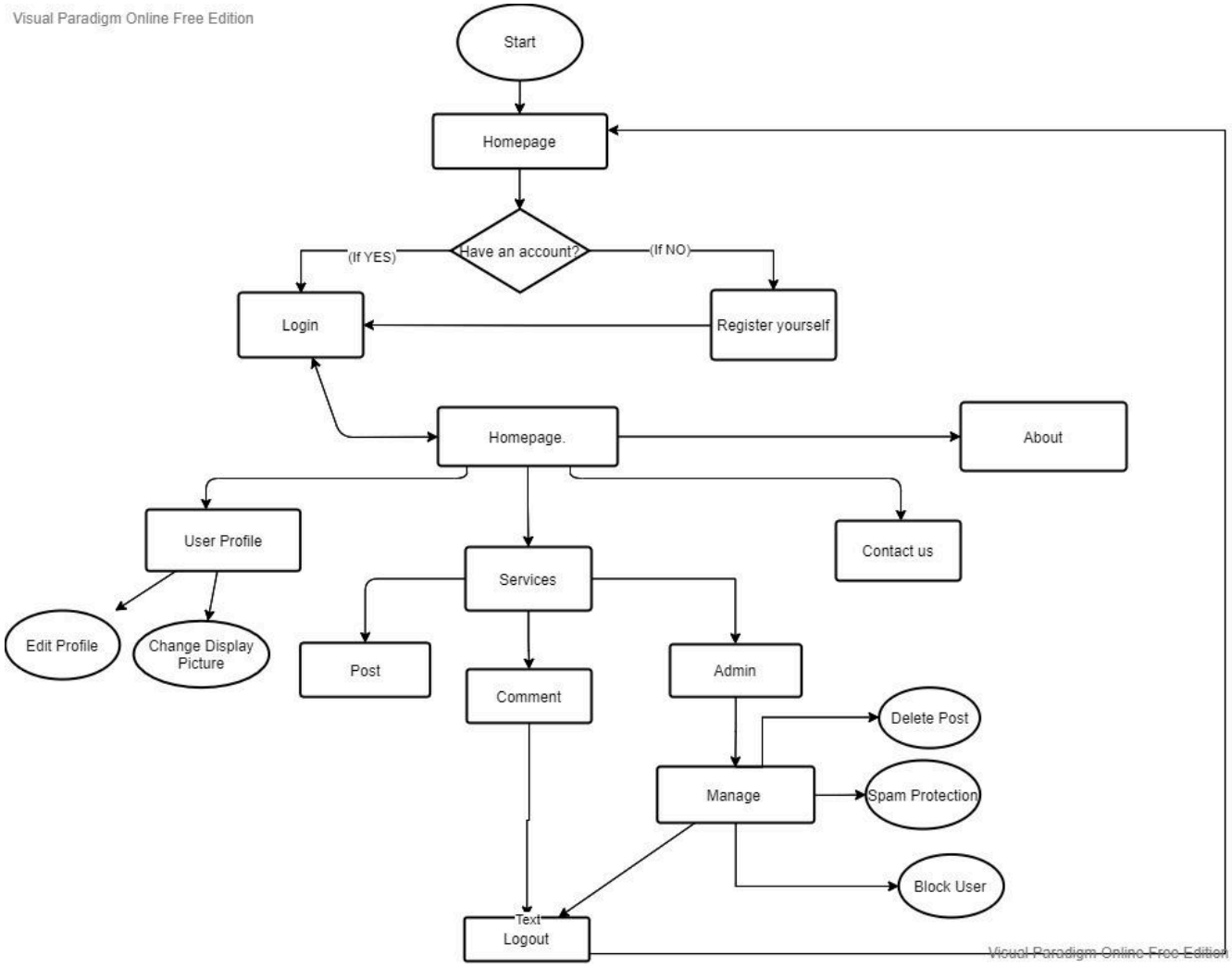
There are many students and lecturers at our gitam university, and it is difficult to locate classmates or teachers on social media without asking them for their profile name on platforms such as Instagram, Twitter, and others. As a student, interacting with staff members or obtaining their contact information is difficult. It's also difficult for us to locate our classmates on social media. Through blogging, our G-Blog allows all gitamites to contribute and receive information about university from academics and other gitamites. We can not only chat to people individually on G-Blog, but we can also post messages, photographs, and videos online for all gitamites to see. It also allows students and professors to express themselves in a secure online environment. This encourages students to participate more actively and interactively in conversations about our university. This online platform allows students to effortlessly express themselves, while also encouraging creativity and self-expression.

This project is from the web development field. Coming to the features of our g-blog-

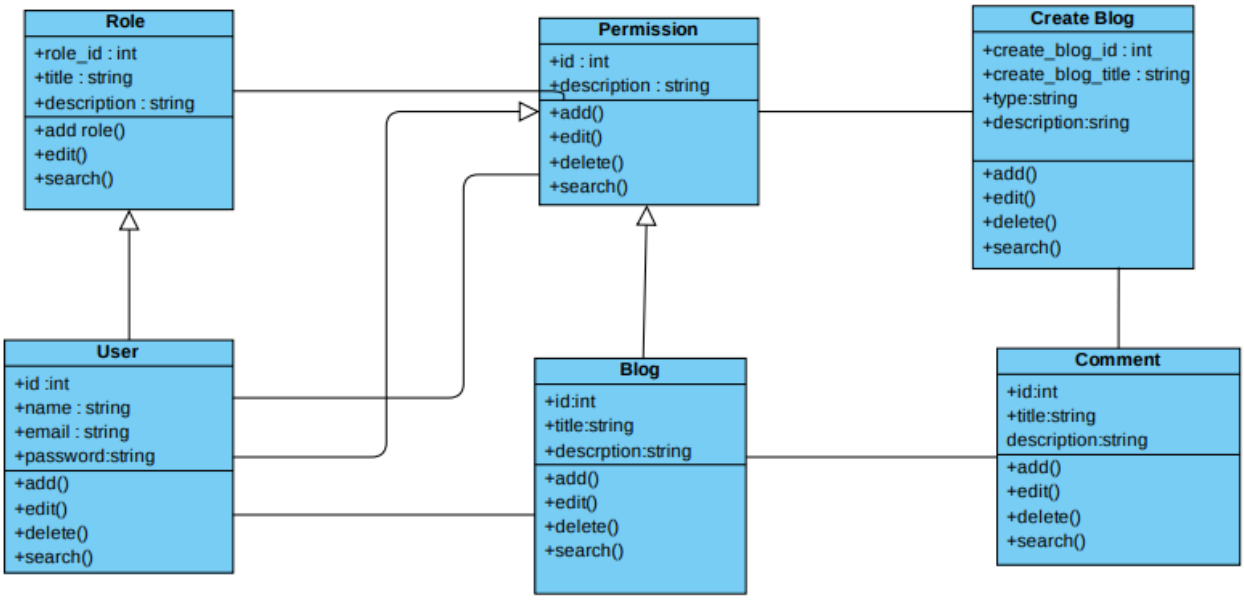
- User accounts where we can see the user’s information.
- Login/sign up where we can login to our account which is already created or sign up and create new accounts .
- Admin panel where the admin can control the activities of the users .
- Add/edit and delete blog posts where users can manage their blog.
- Share posts where users can post messages, pictures or videos.
- Forgot password
- Add/Change/Delete to profile picture

System Methodology

Flow Chart



Class Diagram :



Overview of Technologies

- **Visual Studio Code** - Visual Studio is a Microsoft Integrated Development Environment (IDE) for creating GUIs, consoles, Web applications, online apps, mobile apps, cloud, and web services, among other things. You may write both managed and native code with the help of this IDE. It makes use of Microsoft's different software development platforms, such as Windows Store, Microsoft Silverlight, and Windows API, among others. It is not a language-specific IDE because it can be used to write code in C#, C++, VB(Visual Basic), Python, JavaScript, and a variety of other languages. It is compatible with 36 different programming languages. It's compatible with both Windows and Mac OS X.

Evolution of Visual Studio: Visual Studio 97, with version number 5.0, was the first version of VS(Visual Studio) published in 1997. Visual Studio 15.0 was published on March 7, 2017, and it is the most recent version. Visual Studio 2017 is another name for it. The was backed up. Framework for the Internet Versions range from 3.5 to 4.7 in the most recent version of Visual Studio. Java was supported in previous versions of Visual Studio, however it is no longer supported in the most recent version.

Features

VS Code is compatible with a wide range of programming languages, including Java, C++, Python, CSS, Go, and Dockerfile. VS Code also lets you add on and even create new extensions, such as code linters, debuggers, and support for cloud and web development.

In comparison to other text editors, the VS Code user interface allows for a lot of interactivity. VS Code is separated into five primary regions to make the user experience easier:

- The activity bar
- The sidebar

- Editor groups
- The panel
- The status bar

●**Django** -Django is a high-level Python web framework that allows you to easily create secure and maintainable websites. Django is a web framework created by experienced developers that handles much of the heavy lifting so you can concentrate on designing your project rather than reinventing the wheel. It's open source and free, with a thriving community, comprehensive documentation, and a range of free and paid support options.

Django helps you write software that is:

Complete

Django adheres to the "batteries included" principle, which means it comes with practically everything a developer would need "out of the box." Because everything you need is included in one "product," it all functions together flawlessly, adheres to the same design principles, and comes with comprehensive and up-to-date documentation.

Versatile

From content management systems and wikis to social networks and news sites, Django can (and has) been used to create practically any style of website. It can integrate with any client-side framework and serve material in nearly any format (including HTML, RSS feeds, JSON, XML, etc). Django was used to create the website you're looking at right now!

Internally, it may be modified to use other components as needed, while providing options for practically any capability you might want (e.g., various major databases, templating engines, etc.).

Secure

Django provides a framework that has been engineered to "do the right things" to defend the website automatically, which helps developers avoid many common security blunders. Django, for example, avoids common pitfalls like storing session information in cookies, which is vulnerable (instead, cookies merely hold a key, and the actual data is saved in the database), and directly storing passwords rather than a password hash.

SQL injection, cross-site scripting, cross-site request forgery, and clickjacking are all vulnerabilities that Django protects against by default.

Maintainable

Django programming is created with design principles and patterns in mind, resulting in code that is easy to maintain and reuse. It employs the Don't Repeat Yourself (DRY) philosophy to eliminate needless repetition and hence reduce the amount of code. Django also encourages the grouping of comparable functionality into reusable "applications," as well as the grouping of related code into modules at a lower level

Portable

Django is written in Python, a programming language that runs on a variety of platforms. That means you're not locked into a single server platform and can run your apps on a variety of Linux, Windows, and Mac OS X flavours. Django is also well-supported by a large number of web hosting companies, who often provide dedicated infrastructure and documentation for hosting Django sites.

Where did it come from?

Django was created by a web team that was in charge of establishing and maintaining newspaper websites between 2003 and 2005. After building a few sites, the team started to factor out and reuse a lot

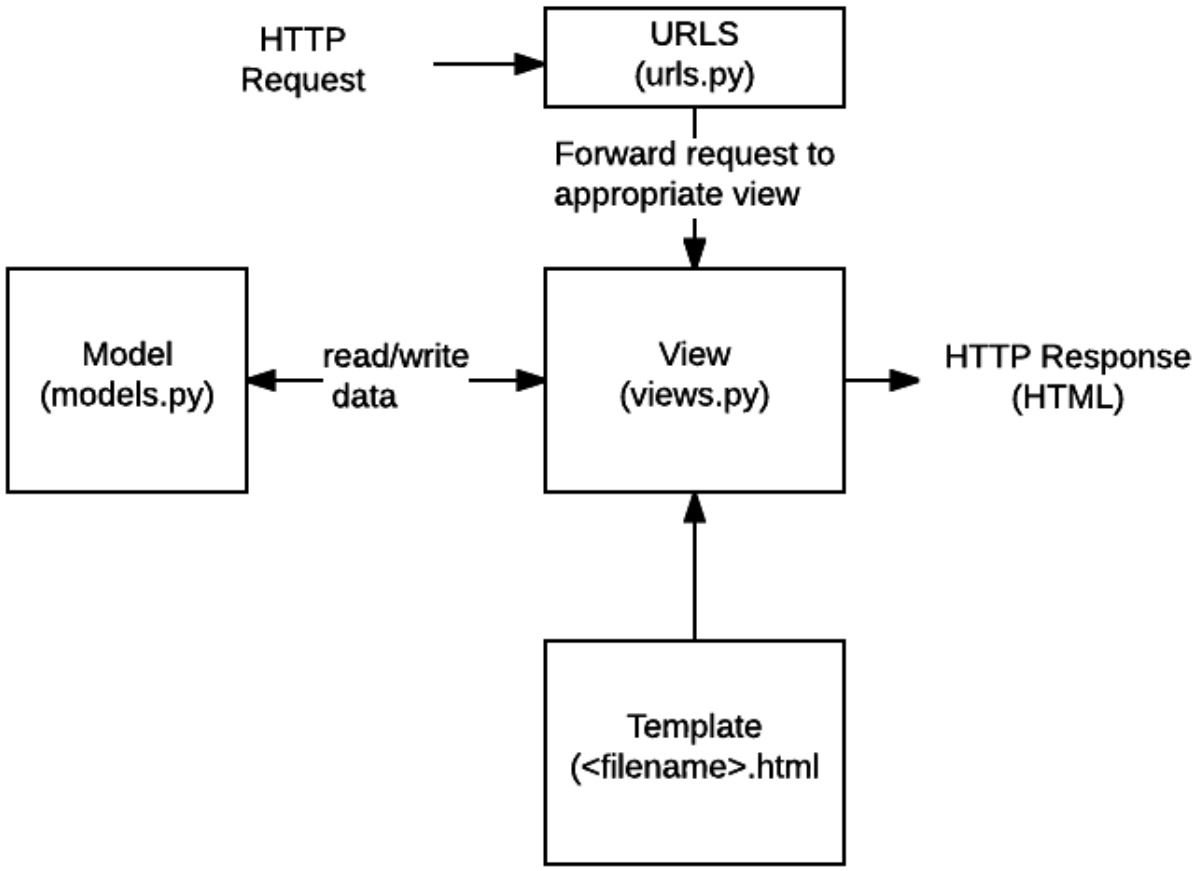
of the same code and design principles. This shared code grew into the "Django" web development framework, which was released as open-source in July 2005. From its first milestone release (1.0) in September 2008 to the most recent version 3.1, Django has continued to evolve and improve (2020). From support for new types of databases, template engines, and caching to the addition of "generic" view functions and classes, each release has provided additional features and bug fixes (which reduce the amount of code that developers have to write for a number of programming tasks).

With thousands of users and contributors, Django is currently a vibrant, collaborative open source project. Django has evolved into a versatile framework capable of producing any form of website. While it still has some characteristics that represent its origins, it has become a framework capable of developing any type of website.

What does Django code look like?

A web application waits for HTTP requests from the web browser in a standard data-driven website (or other client). When an application receives a request, it determines what is required based on the URL and perhaps information in the POST or GET payload. It may next read or write information from a database or do other operations to complete the request, depending on what is necessary. The programme will then send a response to the web browser, commonly by putting the acquired data into placeholders in an HTML template and dynamically constructing an HTML page for the browser to display.

Django web applications typically group the code that handles each of these steps into separate files:



- **URLs:**While it is feasible to process requests from all URLs with a single function, it is significantly more maintainable to write a distinct view function for each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also search for certain patterns of strings or digits in a URL and send that information to a view function as data.

- **View:**A view is an HTTP request handler function that receives and responds to HTTP requests. Models provide views with the data they need to fulfil requests, while templates handle the return formatting.

- **Models:** Models are Python objects that define the data structure of an application and allow database entry management (add, update, delete) and querying capabilities.

- **Templates:** A template is a text file that contains placeholders for actual content and dictates the structure and layout of a file (such as an HTML page). An HTML template can be used by a view to dynamically build an HTML page and populate it with data from a model. A template can be used to specify the structure of any type of file; it isn't need to be HTML!

Sending the request to the right view (urls.py)

The urls.py file is normally where a URL mapper is saved. In the example below, the mapper (urlpatterns) establishes a set of mappings between routes (certain URL patterns) and view functions. If an HTTP Request with a URL matching a defined pattern is received, the view function will be called and the request will be delivered to it.

```
urlpatterns = [
 path('admin/', admin.site.urls),
        path('book/<int:id>/', views.book_detail, name='book_detail'),
 path('catalog/', include('catalog.urls')),
 re_path(r'^([0-9]+)/$', views.best),
]
```

The urlpatterns object is a list of path() and/or re path() functions (Python lists are constructed using square brackets, with items separated by commas and a trailing comma optional). [item1, item2, item3,] for example).

A route (pattern) that will be matched is the first argument to both procedures. Angle brackets are used in the path() method to define sections of a URL that will be captured and provided as named arguments to the view function. The re path() function employs a regular expression, which is a flexible pattern matching strategy.

When the pattern is matched, the second parameter is another function that will be invoked. The notation views.book detail specifies that the method is called book detail() and is located in the views module (i.e. in the views.py file).

Handling the request (views.py)

The heart of the web application is the view, which receives HTTP requests from web clients and responds with HTTP answers. In the meantime, they use the framework's other resources to access databases, render templates, and so on.

The example below demonstrates a simple view function index(), which might have been called by our URL mapper from earlier.

It takes a HttpRequest object as a parameter (request) and returns a HttpResponse object, much like all view functions. We don't do anything with the request in this scenario, and our answer is a hard-coded string. In a later part, we'll show you a request that performs something more interesting.

```
# filename: views.py (Django view functions)


from django.http import HttpResponse


def index(request):
```

```
# Get an HttpRequest - the request parameter

# perform operations using information from the request.

# Return HttpResponse

return HttpResponse('Hello from Django!')
```

Defining data models (models.py)

Models are Python objects that Django web applications use to manage and query data. Models specify the data structure, including field types and possible maximum sizes, default values, selection list options, help text for documentation, label text for forms, and so on. The model is defined independently of the underlying database, which you can select from a list of options in your project settings. You don't need to communicate to the database directly once you've decided which one to use — all you have to do now is write your model structure and other code, and Django will take care of the "dirty work" of talking with the database.

The code sample below shows a very basic Django model for a Team object. The Team class is built using Django class models. Model. It defines the team name and team level as character fields and determines a maximum number of characters to be saved for each entry. We describe the team level as a choice field with a mapping between shown options and data to be stored, as well as a default value, because it can have several values.

```
# filename: models.py



from django.db import models
```

```
class Team(models.Model):

 team_name = models.CharField(max_length=40)



        TEAM_LEVELS = (

  ('U09', 'Under 09s'),

  ('U10', 'Under 10s'),

  ('U11', 'Under 11s'),

  ... #list other team levels

)

 team_level = models.CharField(max_length=3, choices=TEAM_LEVELS, default='U11')
```

Rendering data (HTML templates)

Template systems allow you to define the structure of an output document by inserting placeholders for data that will be filled in when the page is generated. Templates are commonly used to construct HTML, but they can also be used to build other documents. Django comes with support for both its own templating system and Jinja2, a popular Python package (it can also be made to support other systems if needed).

The code sample depicts the HTML template that was called by the render() function in the previous section.

. When this template is rendered, it is assumed that it will have access to a list variable named youngest teams (which is stored in the context variable within the render() function above). We have an

expression inside the HTML skeleton that checks if the youngest teams variable exists before iterating it in a for loop. The template displays each team's team_name value in a li> element on each iteration.

```
## filename: best/templates/best/index.html



<!DOCTYPE html>

<html lang="en">

<head>

 <meta charset="utf-8">

 <title>Home page</title>

</head>

<body>

 {% if youngest_teams %}

  <ul>

   {% for team in youngest_teams %}

    <li>{{ team.team_name }}</li>

   {% endfor %}

  </ul>

 {% else %}

  <p>No teams are available.</p>
```

```
 {% endif %}

</body>

</html>
```

What can we do with django ?

URL mapping, views, models, and templates are the major aspects that you'll utilise in practically every online application, as seen in the prior sections. Django also provides a number of other features, such as:

- **Forms**:HTML Forms are used to collect data from users for server processing. Django makes creating, validating, and processing forms easier.
- **User authentication and permissions**: Django comes with a strong user authentication and authorization mechanism that was designed with security in mind.
- **Caching**: Dynamic content creation is substantially more computationally intensive (and slower) than static content delivery. Django has configurable caching, allowing you to save all or part of a rendered page so that it is only re-rendered when absolutely necessary.
- **Administration site**: When you use the basic skeleton to build an app, the Django administration site is included by default. It's a cinch to provide site administrators access to an admin page where they can create, change, and examine any data models on your site.
- **Serialising data**: Django makes serialising and serving data as XML or JSON simple. This is beneficial when building a web service (a website that only feeds data to other applications or sites and does not display anything), or when building a website where the client-side JavaScript handles all data presentation.

●**SQLite3** -SQLite is a public-domain SQL database engine that is self-contained, high-reliability, embedded, and full-featured. It is the world's most popular database engine. It's a work-in-progress library with open-source code. It can be used for any purpose, whether business or personal. It's essentially a SQL database engine that's integrated. Because SQLite does not have a separate server like SQL, it can read and write ordinary disc files with ease. Because the SQLite database file format is cross-platform, anyone may copy a database between 32-bit and 64-bit platforms with ease.

SQLite is a C library that contains a relational database management system (RDBMS). SQLite is not a client–server database engine, unlike many other database management systems. Rather, it is incorporated into the final product.

The syntax of SQLite is similar to that of PostgreSQL. SQLite makes use of a dynamically typed, weakly typed SQL language.

This syntax does not ensure the domain's integrity. This means that a string can be inserted into an integer column, for example. SQLite will try to convert data between formats where possible, in this case converting the string "123" to an integer, but it cannot guarantee such conversions and will save the data as-is if one is not possible.

SQLite is a popular choice for embedded database software in application software such as web browsers for local/client storage. It is likely the most commonly used database engine, as it is currently used by a number of popular browsers, operating systems, and embedded systems (such as mobile phones). Many programming languages have SQLite bindings.

Design

The SQLite engine, unlike client–server database management systems, has no separate processes with which the application programme interacts. The SQLite library, on the other hand, is linked in and therefore becomes a part of the application programme. Static or dynamic linking is possible. Simple function calls are used by the application software to access SQLite's capabilities, which reduces

database access latency: function calls within a single process are more efficient than inter-process communication.

On a host machine, SQLite saves the complete database (definitions, tables, indices, and data) in a single cross-platform file. This straightforward approach is implemented by locking the entire database file while it is being written. Read operations in SQLite can be multitasked, but writes must be done in order.

SQLite applications require less configuration than client–server databases due to their server-less nature. Because SQLite does not require service administration (such as startup scripts) or access control based on GRANT and passwords, it is referred to as zero-conf. The database file itself is granted file-system permissions, which are used to manage access. In client–server systems, databases use file-system permissions that grant only the daemon process access to the database files.

Another consequence of the serverless architecture is that multiple processes may be unable to write to the database file. Several authors will connect to the same daemon in a server-based database, which will be able to handle the locks internally. On the other hand, SQLite must rely on file-system locking. It is less aware of the other programmes that are concurrently accessing the database. As a result, SQLite is not the best solution for deployments that require a lot of writing. SQLite performance benefits from reducing the burden of sending data to another process for basic queries with little concurrency.

As a reference platform, SQLite leverages PostgreSQL. The phrase "What would PostgreSQL do?" is used to explain the SQL standard. One significant difference is that SQLite does not enforce type checking, with the exception of primary keys; the type of an item is dynamic and not rigorously bound by the schema (although the schema will trigger a conversion when storing, if such a conversion is potentially reversible). SQLite tries to follow Postel's rule as closely as possible.

Features

SQLite implements the majority of the SQL-92 standard, but some functionalities are missing. It can't write to views, for example, and only partially offers triggers (however, it provides INSTEAD OF triggers that provide this functionality). It only supports a few ALTER TABLE statements.

In contrast to most SQL database systems, SQLite has an unusual type system for a SQL-compatible DBMS: instead of assigning a type to a column, types are allocated to individual values; in language terms, it is dynamically typed. Furthermore, it is weakly typed in similar respects to Perl: a string can be inserted into an integer column (though SQLite will try to convert the string to an integer first if the column's preferred type is integer). This gives columns more versatility, especially when they're tied to a dynamically typed programming language. The technique, however, is not transferable to other SQL products.The lack of data integrity afforded by statically typed columns in competing products is a typical critique of SQLite's type system. A "strict affinity" mode is described on the SQLite website, however it has yet to be implemented. It can, however, be achieved using constraints such as CHECK(typeof(x)='integer').

A hidden rowid index column is usually present in tables, allowing for speedier access. SQLite will often optimise an Integer Primary Key column by considering it as an alias for rowid, causing the contents to be stored as a tightly typed 64-bit signed integer and changing its behaviour to that of an auto-incrementing column. To distinguish these columns from weakly typed, non-auto incrementing Integer Primary Keys, future SQLite versions may offer a command to introspect if a column has behaviour similar to rowid.

Optionally, SQLite with a complete Unicode function can be used.

The same database may be accessed by multiple computer processes or threads at the same time. Several read accesses can be completed at the same time. Only if no other accesses are currently being serviced can a write access be satisfied. If this is not the case, the write access will fail with an error code (or can automatically be retried until a configurable timeout expires). When dealing with temporary tables, the scenario with concurrent access would alter. When write-ahead logging (WAL) is enabled, this restriction is lifted in version 3.7, allowing concurrent reads and writes.

Foreign key limitations were implemented in version 3.6.19, which was released on October 14, 2009.

The FTS4 (full-text search) module was added to SQLite version 3.7.4 for the first time, with improvements over the prior FTS3 module.

FTS4 enables users to search full-text documents in the same way that search engines search webpages. Support for generating tables without a rowid was added in version 3.8.2, which could save space and increase speed. In version 3.8.3, SQLite added support for common table expressions. In version 3.8.11, a newer search module named FTS5 was added, with more radical (relative to FTS4) improvements necessitating a version bump.

SQLite version 3.9 offered JSON content management in 2015, with the json1 extension and new subtype interfaces.

The highest database size supported by version 3.33.0 is 281 TB.

●**Json** -JavaScript Object Notation (JSON) is an acronym for JavaScript Object Notation. It's a data-transfer format that's easy to use. It's just plain text written in JavaScript object notation that's used to transfer data between machines. JSON is language agnostic.

JSON is a data interchange format that is lightweight, text-based, and language-independent. It is based on the Javascript/ECMAScript programming language, but it is not dependent on it.According to JSON: The Fat-Free Alternative to XML, a 2006 presentation by JSON's developer, Douglas Crockford, JSON is considered a lightweight alternative to XML. For the portable representation of structured data, JSON defines a modest set of structural conventions. For expressing objects, collections of name/value pairs, and arrays, ordered lists of values, JSON provides a straightforward language. Trees and other complicated data structures can be represented by nesting these two basic types.

JSON grammar arranges data that are strings (sequences of characters in double quotes) or numbers (represented as sequences of digits) into the two basic structures using braces (curly brackets:), square brackets ([]), colons, and commas.

An object is a collection of name/value pairs that are not in any particular order. An item starts with (left brace) and ends with (right brace) (right brace). Each name is preceded by a colon, and name/value pairs are separated by a comma (comma). An array is a collection of values that are arranged in a specific order. [(left bracket)] [(right bracket)] [(right bracket)] [(right bracket)] [(right bracket)] [(right (right

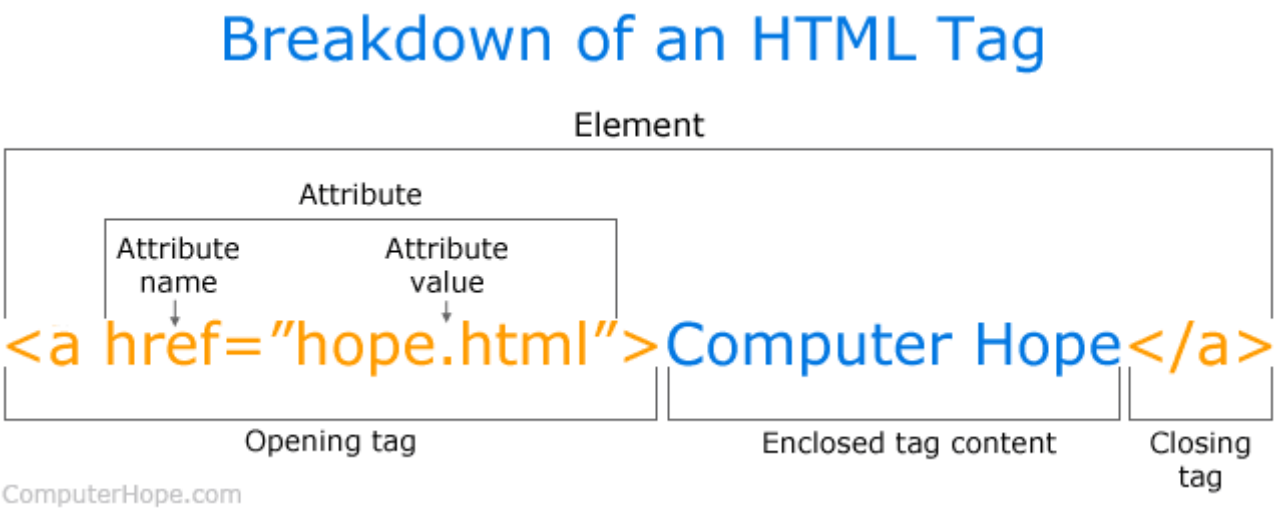
bracket). Values are separated by a comma (,). (comma). A value can be a string enclosed in double quotes, a number, true or false, null, or an object, or an array.These structures can be nested

- **HTML -** HyperText Markup Language is the abbreviation for HyperText Markup Language. It's a markup language that's used to create web pages. HTML is a markup language that combines hypertext with markup. The term "hypertext" refers to the link between web pages. The text document within a tag that defines the structure of web pages is defined using a markup language. This language is used to annotate (add notes to) text so that a computer can understand it and manipulate it appropriately. The majority of markup languages (such as HTML) are readable by humans.

The language uses tags to define what manipulation has to be done on the text.

What does an HTML tag look like?

There aren't many components, as illustrated in the HTML tag example above. Most HTML tags have an opening tag that contains the tag name and tag attributes, followed by a closing tag that contains a forward slash and closes the tag name. It is excellent practise to finish tags that do not have a closing tag, such as img>, with a forward slash.



The majority of tags are enclosed in angle brackets of less than and greater than, and everything between the open and close tags is shown or influenced by the tag. The a> tag in the example above creates a link called "Computer Hope" that points to the hope.html page.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
        "https://www.w3.org/TR/html4/loose.dtd">

<html>

 <head>

 <title>Example page</title>

 <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">

 </head>

 <body>

 <h1>This is a heading</h1>

 <p>This is an <b>example</b> of a basic HTML page.</p>

         </body>

         </html>
```

The box above contains the key ingredients to a basic web page. Each of the lines are explained below in further detail.

1. The DOCTYPE line specifies the HTML version used to create the page, allowing an Internet browser to interpret the text that follows.
2. The <html> opening tag lets the browser know that it is reading HTML code.

3. The head> part of the page contains information about the page, such as the title, meta tags, and the location of the CSS file.

4. The <body> section contains everything that's viewable on the browser. For example, all the text seen here is contained in the body tags.

5. The <h1> tag is the visible heading of the page.

6. The <p> tag is a paragraph of text. Most web pages (like this one) have several paragraph tags.

7. The b> tag, which bolds the word example in the paragraph, is included in the text.

8. Finally, the closing tags wrap each of the above tags.

●**CSS** - CSS, or Cascading Style Sheets, is a simple design language intended to make the process of making web pages presentable easier. Styles can be applied to web pages using CSS. More crucially, CSS allows you to do so without having to worry about the HTML code that makes up each web page. CSS is simple to learn and understand, but it gives you a lot of power over how an HTML document looks.

The example box below shows how to use CSS code to determine typefaces, hyperlink colours, and the colour of a link when the mouse pointer hovers over it. We're only modifying the HTML tags a> and body> in this example, and we're not adding any new class or id selectors.

```
body {
 font: normal 100% "trebuchet ms", Arial, Helvetica, sans-serif;
}
a {
 color: #000000;
}
```

```
a:visited {
 color: #005177;
}
a:hover {
 color: #005177;
}
```

The following code can be used to insert the CSS code from the box above into the head section of a page's HTML. Keep in mind, though, that this action only applies the changes to a single page.

```
<style type="text/css">
<!--
 above code inserted here-->
        </style>
```

We recommend storing the CSS code in a separate CSS file and loading it on each page if you want to use it on multiple pages. The CSS code in the first box on this page, for example, can be copied and pasted into a file with the.css extension.

The file must be linked to in the head of the HTML code using the link> element after it has been saved. An example of this element in action is shown in the box below

```
<link rel="stylesheet" Type="text/css" href="URL or path to css file here">
```

The following line would connect the CSS file if it was named example.css and was in the same directory as the HTML file it was loaded from.

```
<link rel="stylesheet" Type="text/css" href="example.css">
```

- **Python** -Python is a high-level, general-purpose programming language that is widely used. The Python programming language (latest Python 3) is utilised in web development, machine learning applications, and all cutting-edge software technology. Python is an excellent programming language for beginners as well as experienced programmers who have worked with other programming languages such as C++ and Java.

We used python-related packages and managers in this project. The python technologies we used are

Pip

What is pip?

Python's standard package manager is pip. It enables you to install and manage packages that aren't included in the Python standard library.

It's a programme that lets you install and manage extra libraries and dependencies that aren't included in the standard library.

Pip has been provided with the Python installer since versions 3.4 for Python 3 and 2.7.9 for Python 2, and it's used by a lot of Python projects, thus it's a must-have tool for any Pythonista.

If you're coming from another language, you might be familiar with the concept of a package manager. For package management, JavaScript utilises npm, Ruby uses gem, and.NET uses NuGet. Pip has become the de facto package manager in Python.

Unless you installed an older version of Python, the Python installer install pip, so it should be ready to use. By executing $ pip —version, you may check if pip is installed.

Python is a battery-powered programming language. This implies that the Python standard library comes with a large number of packages and modules to aid programmers with their scripts and applications

Simultaneously, Python has a vibrant community that contributes an even larger number of packages to aid you in your development efforts. The Python Package Index, or PyPI, is where these packages are released (pronounced Pie Pea Eye). PyPI contains a large number of packages, such as development frameworks, tools, and libraries.

Many of these packages make Python programming easier by offering user-friendly interfaces to functions found in the standard library.

Pipenv

Pipenv is a tool that seeks to bring the best of all worlds of packaging to the Python world (bundler, composer, npm, cargo, yarn, and so on). In our world, Windows is a first-class citizen.

It generates and manages a virtualenv for your projects automatically, as well as adds and removes packages from your Pipfile as you install and uninstall packages. It also creates the crucial Pipfile.lock file, which is required for deterministic builds.

Pipenv is attempting to solve a variety of issues:

- You no longer need to use `pip` and `virtualenv` separately. They work together.
- Because managing a requirements.txt file can be difficult, Pipenv instead relies on the future Pipfile and Pipfile.lock, which are better for basic use cases..
- Hashes are used all the time, everywhere. Security. Vulnerabilities in security are automatically exposed..
- Give you insight into your dependency graph (e.g. `$ pipenv graph`).
- Streamline development workflow by loading `.env` files.

Features

- Enables truly deterministic builds, while easily specifying only what you want.

- Generates and checks file hashes for locked dependencies.
- Automatically install required Pythons, if pyenv is available.
- Automatically finds your project home, recursively, by looking for a Pipfile.
- Automatically generates a Pipfile, if one doesn't exist.
- Automatically creates a virtualenv in a standard location.
- Automatically adds/removes packages to a Pipfile when they are un/installed.
- Automatically loads .env files, if they exist.

Install, uninstall, and lock, which creates a Pipfile.lock, are the main commands. These are meant to take the role of $ pip install and manual virtualenv maintenance (execute $ pipenv shell to activate a virtualenv).

Basic Concepts

- A virtualenv will automatically be created, when one doesn't exist.
- When no parameters are passed to install, all packages [packages] specified will be installed.
- To initialize a Python 3 virtual environment, run $ pipenv --three.
- To initialize a Python 2 virtual environment, run $ pipenv --two.
- Otherwise, whatever virtualenv defaults to will be the default.

Other Commands

- shell will spawn a shell with the virtualenv activated.

- run will run a given command from the virtualenv, with any arguments forwarded (e.g. $ pipenv run python).
- check asserts that PEP 508 requirements are being met by the current environment.
- graph will print a pretty graph of all your installed dependencies.

●**JavaScript** - JavaScript is a scripting language that is lightweight, cross-platform, and interpreted. It is well-known for web page creation, but it is widely used in numerous non-browser applications. JavaScript can be used to create both client-side and server-side applications. A standard library of objects, such as Array, Date, and Math, as well as a basic set of language components, such as operators, control structures, and statements, are included in JavaScript.

Implementation

Coding:-

Python version used

```
D:\gblog>python --version
Python 3.9.1
```

creating a virtual environment

Here we use pipenv to create virtual environment

```
D:\gblog>pipenv shell
Creating a virtualenv for this project...
Pipfile: D:\gblog\Pipfile
Using C:/Users/sdkpr/AppData/Local/Programs/Python/Python39/python.exe (3.9.1) to create virtualenv...
[ ===] Creating virtual environment...created virtual environment CPython3.9.1.final.0-64 in 3024ms
  creator CPython3Windows(dest=C:\Users\sdkpr\.virtualenvs\gblog-EwVqMAXz, clear=False, no_vcs_ignore
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_
    added seed packages: pip==21.3.1, setuptools==58.3.0, wheel==0.37.0
  activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonAc

Successfully created virtual environment!
Virtualenv location: C:\Users\sdkpr\.virtualenvs\gblog-EwVqMAXz
Launching subshell in virtual environment...
Microsoft Windows [Version 10.0.19042.1288]
(c) Microsoft Corporation. All rights reserved.
```

This will start a shell in a virtual environment to separate the application's development.

If a virtual environment does not already exist, this will build one. All of your virtual environments are created at a default place by Pipenv. There are certain environmental variables to configure if you want to change Pipenv's default behaviour.

Now we need to install django with pip install django, but we need to provide the version with == versionname, i.e. pip install django==2.1.

```
(gblog-EwVqMAXz) D:\gblog>pip install django==2.1
Collecting django==2.1
  Using cached Django-2.1-py3-none-any.whl (7.3 MB)
Collecting pytz
  Using cached pytz-2021.3-py2.py3-none-any.whl (503 kB)
Installing collected packages: pytz, django
Successfully installed django-2.1 pytz-2021.3
```

Django is a high-level Python web framework for building secure and maintainable websites quickly. Django is a web framework built by experienced developers that takes care of a lot of the heavy lifting so you can focus on developing your app instead of reinventing the wheel. It's free and open source, with a vibrant and active community, excellent documentation, and a variety of free and paid support options.

Installation of django crispy forms

```
(gblog-EwVqMAXz) D:\gblog>pip install django-crispy-forms==1.7.2
Collecting django-crispy-forms==1.7.2
  Using cached django_crispy_forms-1.7.2-py2.py3-none-any.whl (105 kB)
Installing collected packages: django-crispy-forms
Successfully installed django-crispy-forms-1.7.2
```

Django-crispy-forms is a Python application for managing Django forms. It allows you to change the properties of forms on the backend (such as the method, send button, or CSS classes) without having to rewrite them in the template. This technique is far more readable than normal Django forms, and it makes producing the form template quite simple — it can even be done with only one line of code. Furthermore, django-crispy-forms can change the form's design and style by adding css classes to the entire form, as well as individual fields, labels, and buttons.

Installation of Pillow

```
(gblog-EwVqMAXz) D:\gblog>pip install Pillow==8.0.0
Collecting Pillow==8.0.0
  Using cached Pillow-8.0.0-cp39-cp39-win_amd64.whl (2.1 MB)
Installing collected packages: Pillow
Successfully installed Pillow-8.0.0
```

Pillow is a Python Imaging Library (PIL) that lets you open, modify, and save images in the Python programming language. A wide range of file formats can be recognised and read in the present version. On purpose, write support is restricted to the most commonly used interchange and presentation formats.

From PIL module we import the image class

```
from PIL import Image
```

The Image.open() method reads the image file. Pillow can read over 30 different file formats.

```
    super(Profile, self).save(*args, **kwargs)

    img = Image.open(self.image.path)
```

Installation of asgiref

```
(gblog-EwVqMAXz) D:\gblog>pip install asgiref==3.4.1
Collecting asgiref==3.4.1
  Using cached asgiref-3.4.1-py3-none-any.whl (25 kB)
Installing collected packages: asgiref
Successfully installed asgiref-3.4.1
```

The asgiref.sync module contains two wrappers that allow you to switch between asynchronous and synchronous code at will while taking care of the kinks.

Installing sqlparse

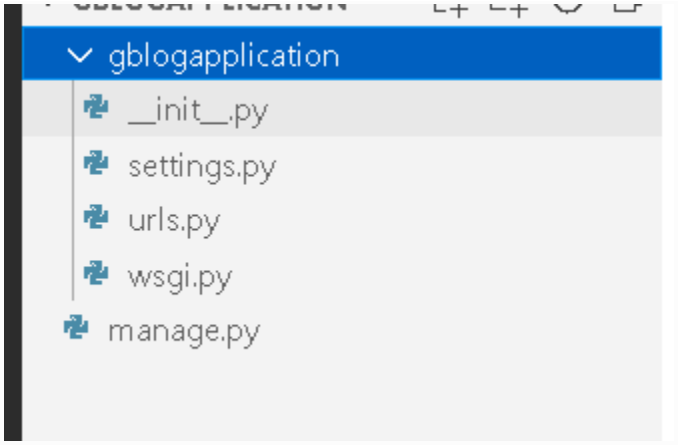
```
(gblog-EwVqMAXz) D:\gblog>pip install sqlparse==0.4.2
Collecting sqlparse==0.4.2
  Using cached sqlparse-0.4.2-py3-none-any.whl (42 kB)
Installing collected packages: sqlparse
Successfully installed sqlparse-0.4.2
```

Sqlparse is a Python SQL parser that does not validate the data. Parsing, dividing, and formatting SQL statements are all supported.

Starting a project

```
(gblog-EwVqMAXz) D:\gblog>django-admin  startproject gblogapplication
```

It will create the following directories and files



- manage.py: This command-line tool allows you to interact with the Django project in a variety of ways. All of the details concerning manage.py can be found in django-admin and manage.py.
- The inner mysite/ directory is the actual Python package for your project. Its name is the Python package name you'll need to use to import anything inside it (e.g. gblogapplication.urls).

- gblogapplication/ init .py: An empty file that instructs Python to treat this directory as a Python package. If you're new to Python, check out the official Python documentation for more information on packages.

- gblogapplication/settings.py: This Django project's settings and setup. Django settings will explain how settings operate in Django.

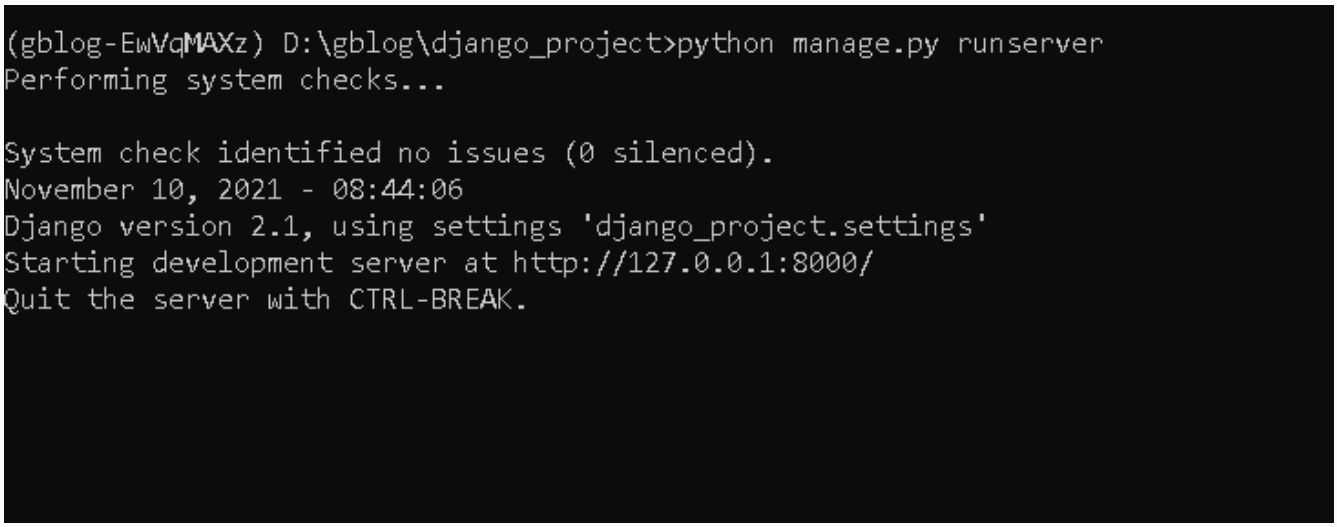
- gblogapplication/urls.py: This Django project's URL declarations; a "table of contents" for your Django-powered site. URL dispatcher has further information on URLs.

- gblogapplication/asgi.py:Your project's entry point for ASGI-compatible web servers. For further information, see How to Deploy with ASGI.

- gblogapplication/wsgi.py: Your project's entry point for WSGI-compatible web servers. For further information, see How to Deploy with WSGI.

Starting a server

To start a server we need to run the following command



This will make the webpage accessible on a local machine.

The runserver command, by default, starts the development server on port 8000 on the internal IP address.

Pass the server's port as a command-line option if you want to change it. This command, for example, launches the server on port 8080:

```
$ python manage.py runserver 8080
```

Pass the server's IP along with the port if you want to alter it. Use the following command to listen on all available public IPs (important if you want to show off your work on other machines on the network):

```
$ python manage.py runserver 0:8000
```

When needed, the development server reloads Python code for each request. The server does not need to be restarted for code modifications to take effect. However, some actions, such as adding files, do not cause the server to restart, so you'll have to do so manually.

Templates

Django templates are text documents or Python strings that have been marked up with the Django template language. The template engine recognises and interprets some constructs. Variables and tags are the most important.

A context is used to render a template. Rendering executes tags and replaces variables with their values, which are looked up in the context. Everything else is output in its current state.

The Django template language has four constructs in its syntax.

```
blog -> templates -> blog -> template.html
```

Above is the convention on where and how the template files should be

views.py

A view function is a Python function that takes a Web request and delivers a Web response, according to the Django documentation. This response could be the HTML content of a Web page, a redirect, a 404 error, an XML document, or an image, or anything else that a web browser can display.

```
def about(request):
    return render(request, 'blog/about.html', {'title': 'About'})
```

The above snippet from views.py when gets a request for about page it goes to the templates and returns the rendered about.html page that is stored in the templates folder inside the blog application.

urls.py

```
urlpatterns = [
    path('', PostListView.as_view(), name='blog-home'),
    path('user/<str:username>', UserPostListView.as_view(), name='user-posts'),
    path('post/<int:pk>/', PostDetailView.as_view(), name='post-detail'),
    path('post/new/', PostCreateView.as_view(), name='post-create'),
    path('post/<int:pk>/update/', PostUpdateView.as_view(), name='post-update'),
    path('post/<int:pk>/delete/', PostDeleteView.as_view(), name='post-delete'),
    path('about/', views.about, name='blog-about'),
]
```

django app

A submodule of the project is referred to as an app. It's self-contained and not intertwined with the other apps in the project, so you could theoretically pick it up and drop it into another project with no changes. In most cases, an app has its own models. python (which might actually be empty). You might consider it an independent Python module. One app may be all that is required for a small project.

If we use templates in a new app, we must add our (blog)app to the list of installed applications so Django knows where to look for templates.

To complete the process, we must now add our app configuration to the project's settings.

INSTALLED APPS = py INSTALLED APPS = py INSTALLED APPS = py INSTALLED

```
# Application definition

INSTALLED_APPS = [
    'blog.apps.BlogConfig',
    'users.apps.UsersConfig',
    'crispy_forms',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

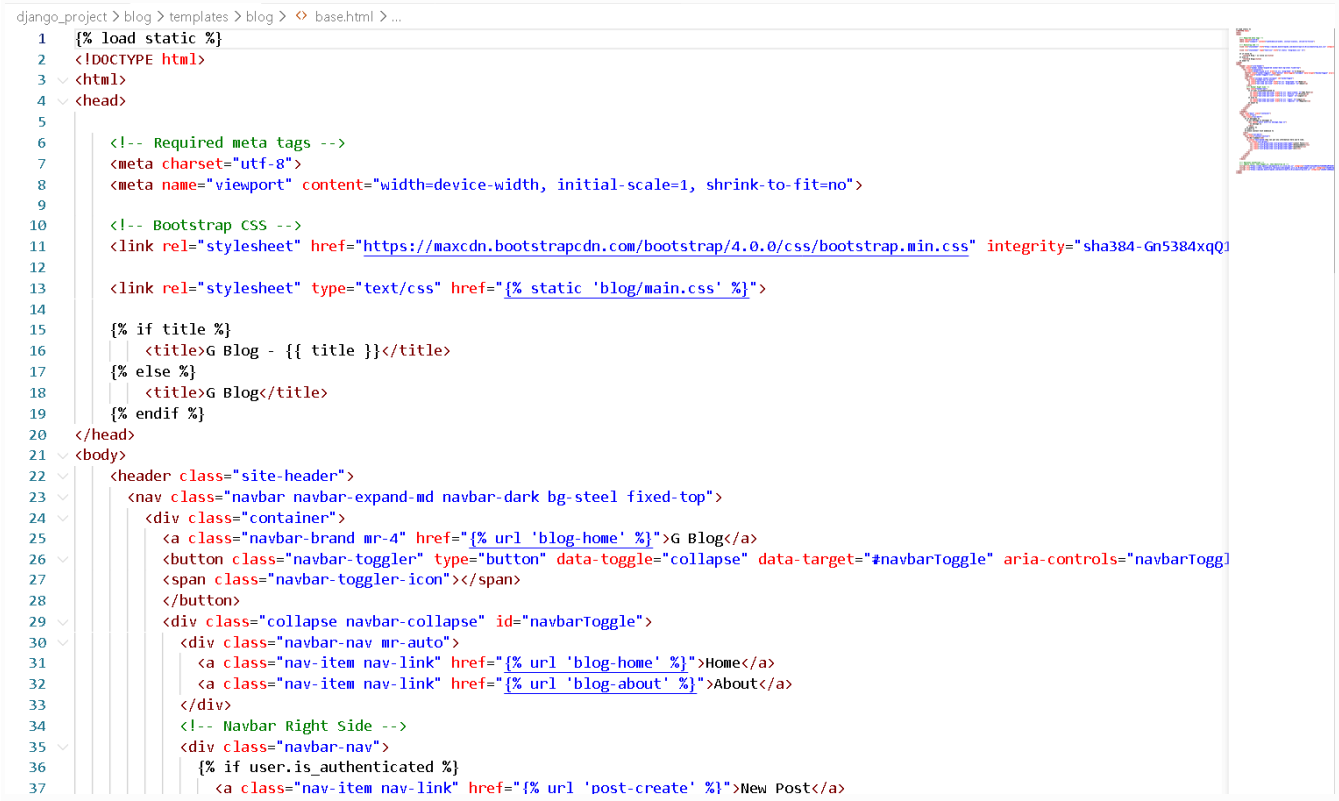
updating settings.py INSTALLED_APPS will help django where to look for the app templates

Template inheritance-

To make other templates inherit some of the characteristics of another template, reducing code complexity and making it easier to maintain.

We'll observe similarities between the about and homepage pages, therefore we'll create a new template and call it basetemplate, i.e. base.html.

If we were not to use template inheritance we would rewrite the whole file of base.html and append to it.



Another advantage of this is if we were to change something we can directly change base.html instead of all the files.

Including the CSS File

We'll build a new folder in our app folder if we want to include a CSS file in django. Static files, such as CSS and JavaScript, must be placed in a static folder.

We should make another folder with the name of our app inside the static folder, and static files should be kept there, just like a template convention.

To include the css file in the base html/template, we must first load our static files, which may be done at the start of base.html with the command percent load static percent.

```
{% load static %}
<!DOCTYPE html>
<html>
<head>

    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content='

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="ht
```

That is why we include {% load static %} atop of base.html and to refer to the css file we do this.

```
<link rel="stylesheet" type="text/css" href="{% static 'blog/main.css' %}">
```

creating superuser

We'll need a user account with the Staff status enabled to access the admin site. This user must also have permissions to control all of our objects in order to see and create data. Using manage.py, you may establish a "superuser" account with complete access to the site and all necessary permissions.

To create the superuser, run the following script in the same directory as manage.py. You'll be asked to create a username, email address, and a secure password.

```
PS E:\gblog\django_project> python .\manage.py createsuperuser
Username (leave blank to use 'nietzschious'): sdkpsuperuser
Email address: 1218103010006@gitam.in
Password:
Password (again):
This password is too common.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

python object relational mapper

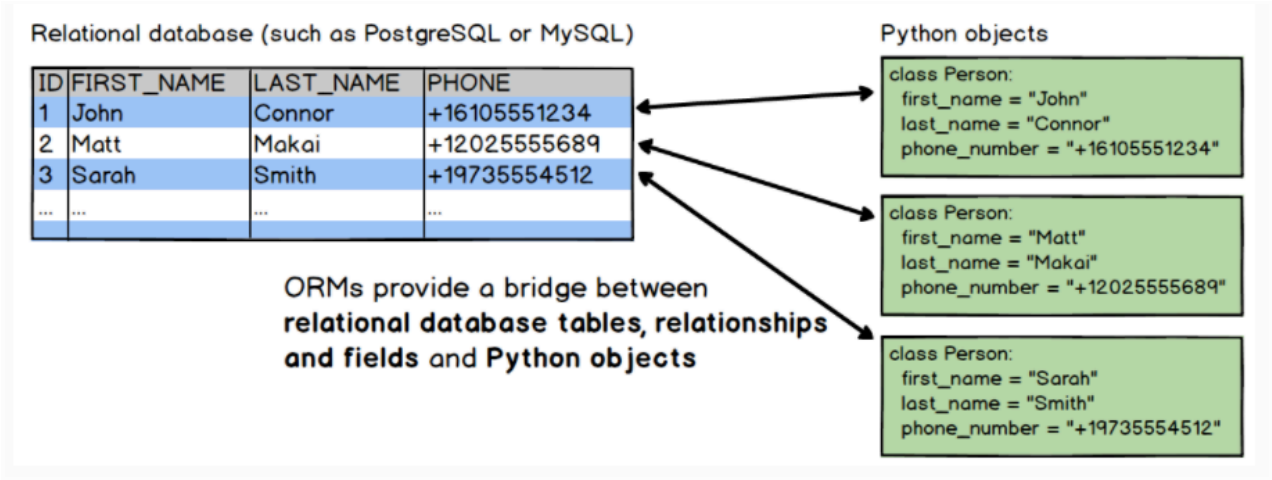
An object-relational mapper (ORM) is a library of code that automates the conversion of data from relational database tables to objects that may be utilised in application code.

ORMs give a high-level abstraction over a relational database, allowing a developer to create, read, update, and delete data and schemas in their database using Python code rather than SQL. Instead of writing SQL statements or stored procedures, developers can deal with a database using their preferred programming language.

```python
class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()
    date_posted = models.DateTimeField(default=timezone.now)
    author = models.ForeignKey(User, on_delete=models.CASCADE)

    def __str__(self):
        return self.title

    def get_absolute_url(self):
        return reverse('post-detail', kwargs={'pk': self.pk})
```



csrf token

The CSRF (Cross-Site Request Forgery) attack causes authenticated users to send a request to a Web application to which they have previously been granted access. CSRF attacks take advantage of a logged-in user's trust in a Web application. Cross-site scripting (XSS) attacks, on the other hand, take advantage of a user's trust in a Web application.) A CSRF attack takes advantage of a flaw in a Web application that can't detect the difference between a request made by a specific user and one made without their knowledge.

An attacker's goal in a CSRF attack is to compel the user to make a state-changing request.

Examples include:

- Submitting or deleting a record.
- Submitting a transaction.
- Purchasing a product.
- Changing a password.
- Sending a message.

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
    <div class="content-section">
      <div class="media">
        <img class="rounded-circle account-img" src="{{ user.profile.image.url }}">
        <div class="media-body">
          <h2 class="account-heading">{{ user.username }}</h2>
          <p class="text-secondary">{{ user.email }}</p>
        </div>
      </div>
      <form method="POST" enctype="multipart/form-data">
          {% csrf_token %}
          <fieldset class="form-group">
              <legend class="border-bottom mb-4">Profile Info</legend>
              {{ u_form|crispy }}
              {{ p_form|crispy }}
          </fieldset>
          <div class="form-group">
              <button class="btn btn-outline-info" type="submit">Update</button>
          </div>
      </form>
    </div>
```

Django defends against cross-site request forgery (CSRF) attacks by generating a CSRF token on the server, sending it to the client, and requiring the client to give the token back in the request header. The server will then check if the token provided by the client matches the one produced previously; if it does not, the request will be denied.

Forms.py

To keep your code easily maintained, the django manual recommends putting all your forms code in forms.py. Additionally, because it is a documented convention, it helps when collaborating with others

because that is where others will expect to find your form-related code.

```
class UserRegisterForm(UserCreationForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']


class UserUpdateForm(forms.ModelForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email']


class ProfileUpdateForm(forms.ModelForm):
    class Meta:
        model = Profile
        fields = ['image']
```

email and password reset

The SMTP host and port supplied in the EMAIL HOST and EMAIL PORT parameters are used to send mail. If set, the EMAIL HOST USER and EMAIL HOST PASSWORD settings authenticate to the SMTP server, while the EMAIL USE TLS and EMAIL USE SSL options govern whether a secure connection is used.

We require an email server to send emails on our behalf.

There are numerous alternative methods for sending an email.

Here, we used Gmail.

```
LOGIN_URL = 'login'

EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 587
EMAIL_USE_TLS = True
EMAIL_HOST_USER = os.environ.get('GITAM_EMAIL_USER')
EMAIL_HOST_PASSWORD = os.environ.get('EMAIL_PASS')
```

Here The environment variables GITAM EMAIL USER and EMAIL PASS are used.

We can directly enter our email account and password here, but this is not suggested; instead, we should save those credentials locally and access them from there as well as the local.

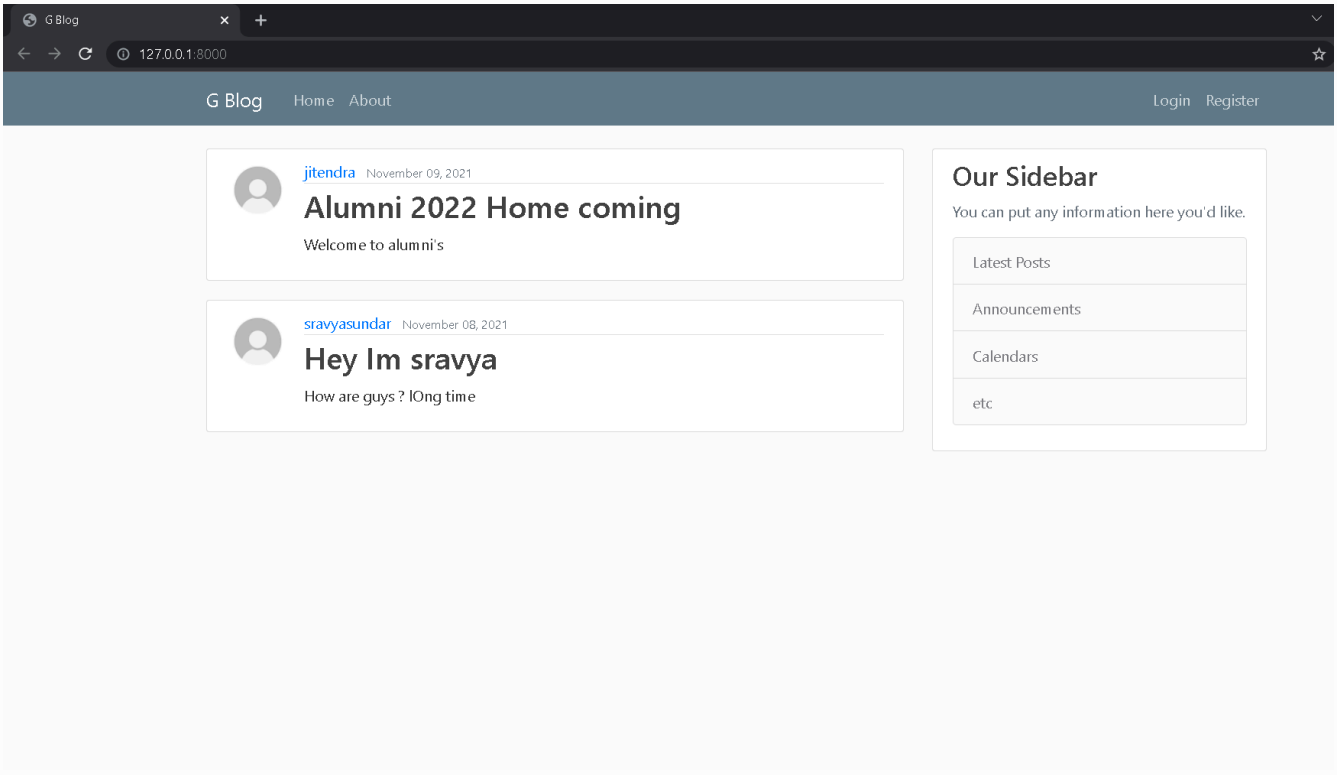
To do so, however, we must first hide passwords and keys saved on our local workstation.

running the server

```
PS E:\gblog\django_project> python .\manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
November 10, 2021 - 15:20:31
Django version 2.1, using settings 'django_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

We can access the webpage now with the help of the ipaddress and the port it was hosted on.



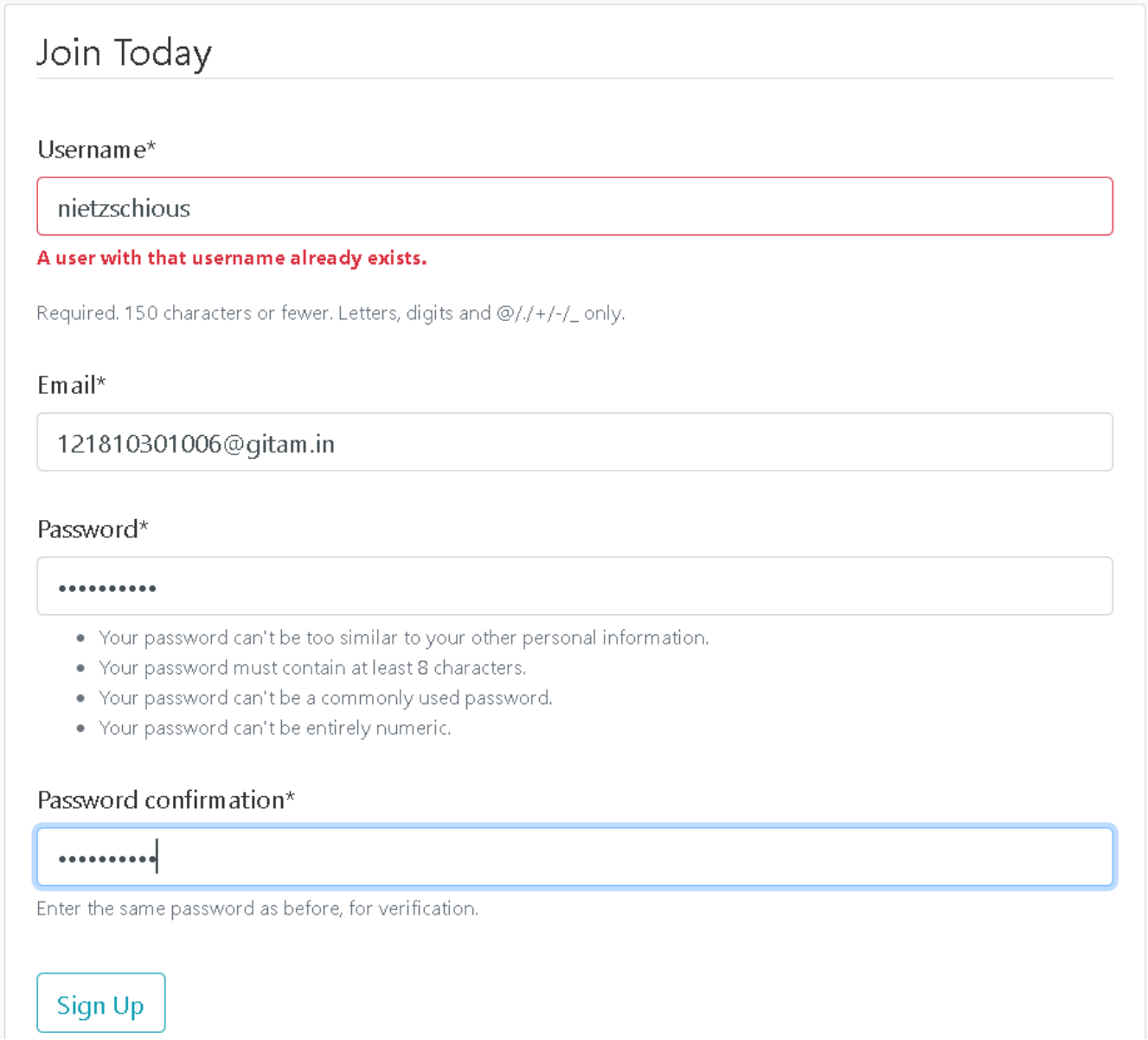
We can see the get request below

```
PS E:\gblog\django_project> python .\manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
November 10, 2021 - 15:20:31
Django version 2.1, using settings 'django_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[10/Nov/2021 15:21:16] "GET / HTTP/1.1" 200 4198
```

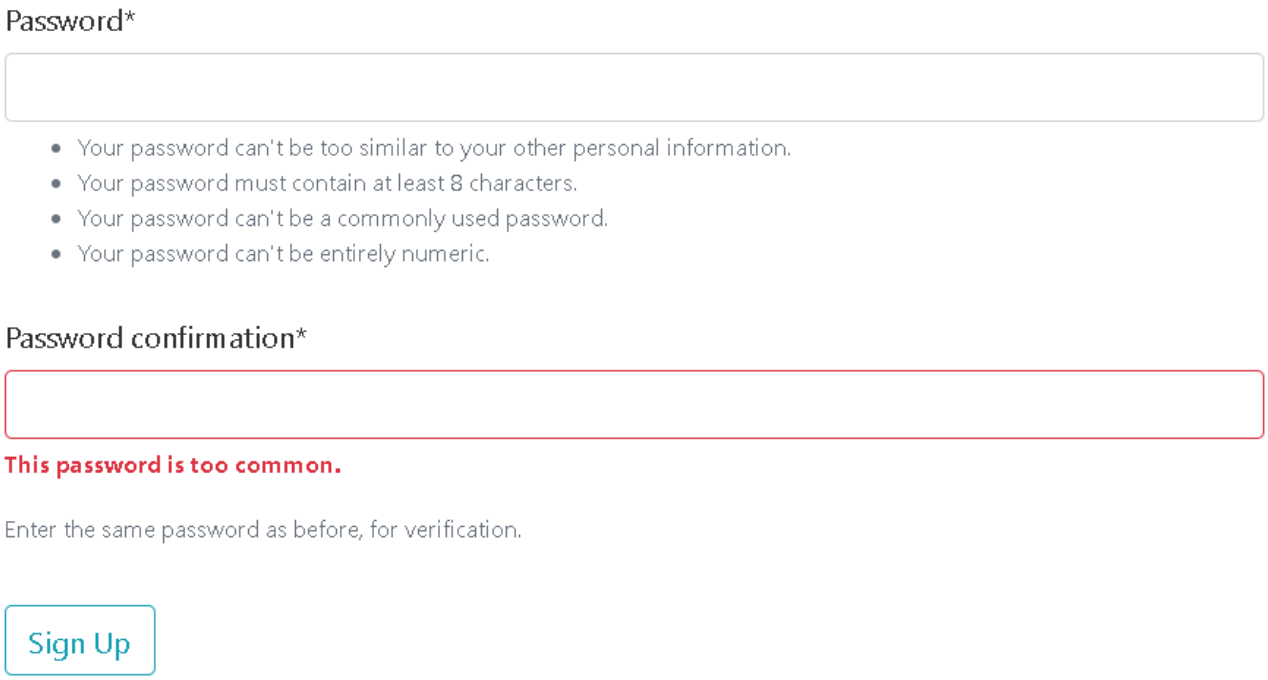
Testing:-

Testcase1 same username while registering validation

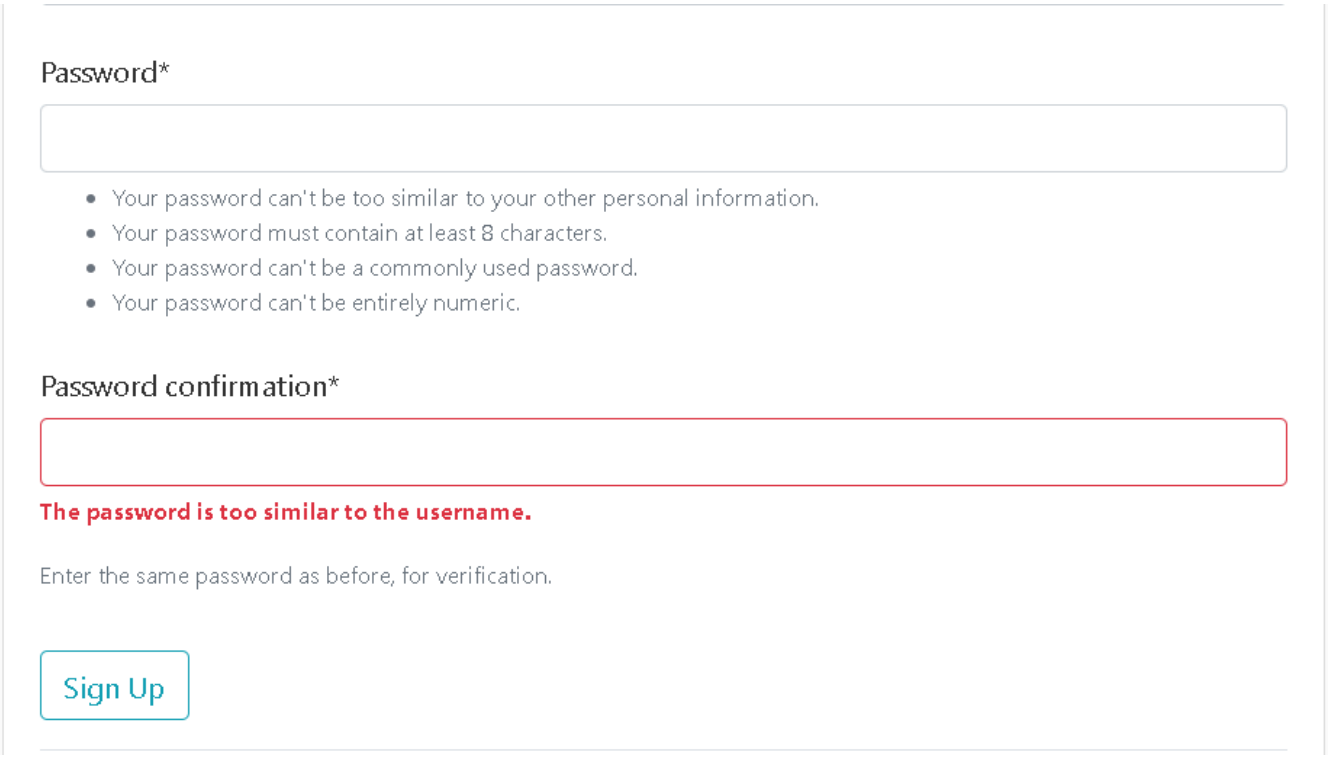


Testcase 2 checking for a valid email address

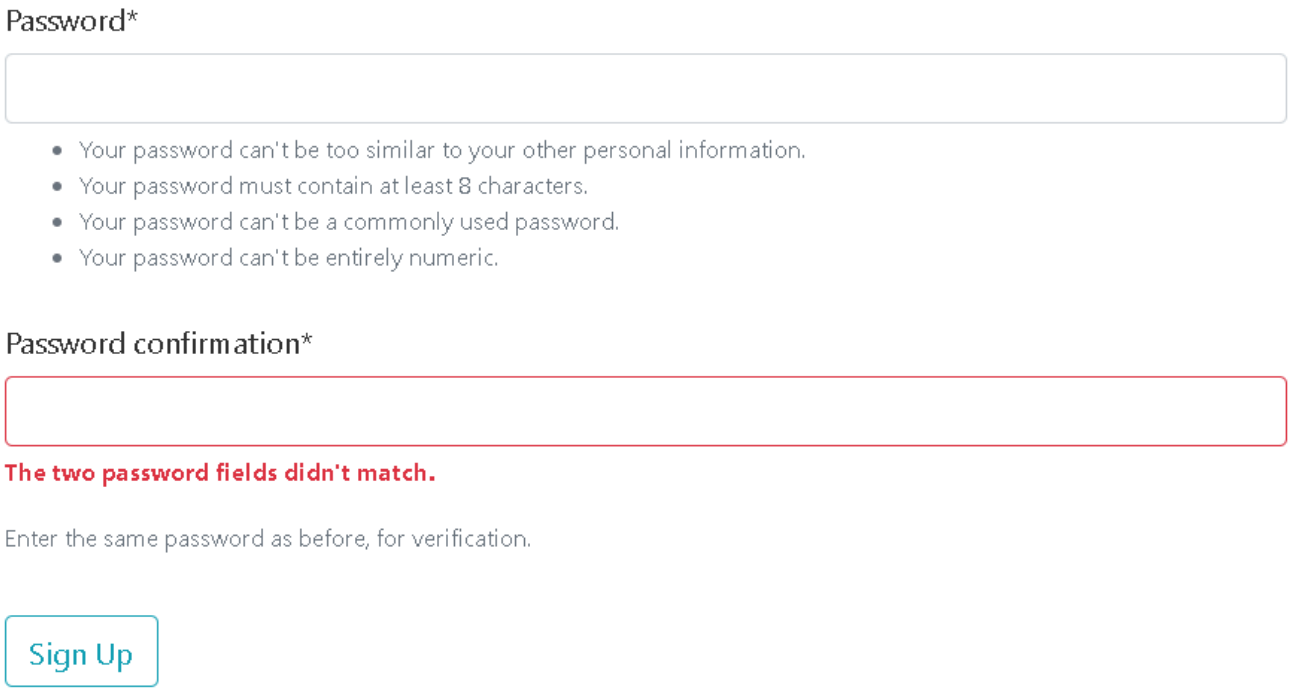
Testcase 3 too common passwords validation



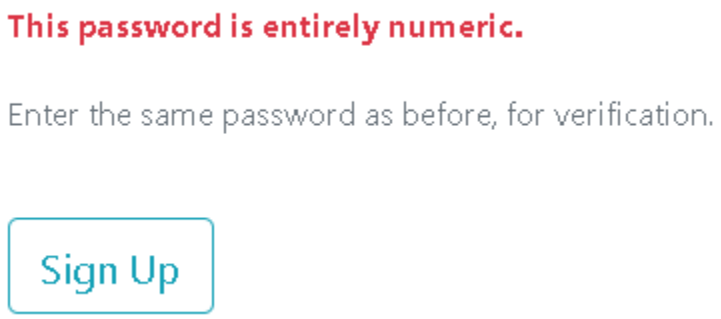
Testcase 4 username and password similar alert



Testcase 5 password and confirm password didn't match



Testcase 6 password entirely numeric validation



Testcase 7 reporting creation of user

Join Today

Username*

helloworld

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email*

121810301006@gitam.in

Password*

••••••••

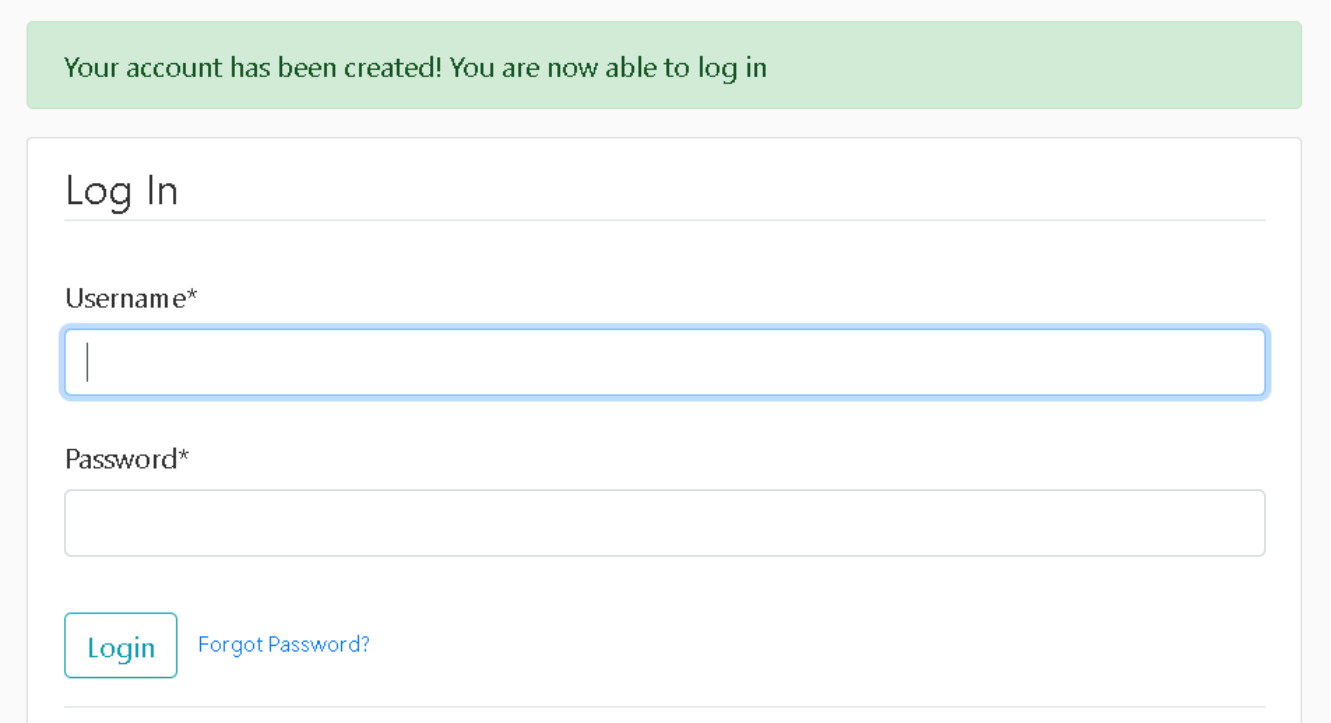
- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation*

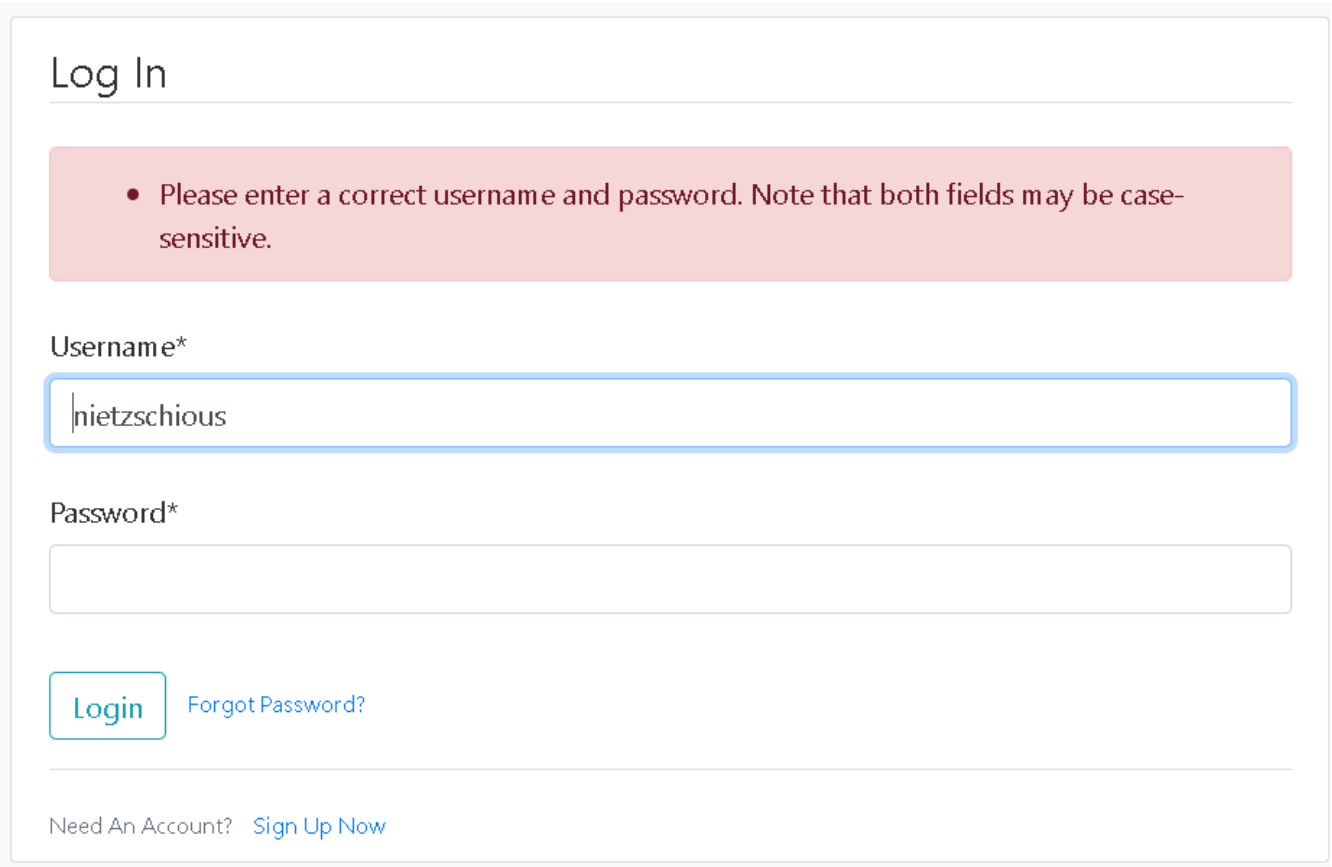
••••••••

Enter the same password as before, for verification.

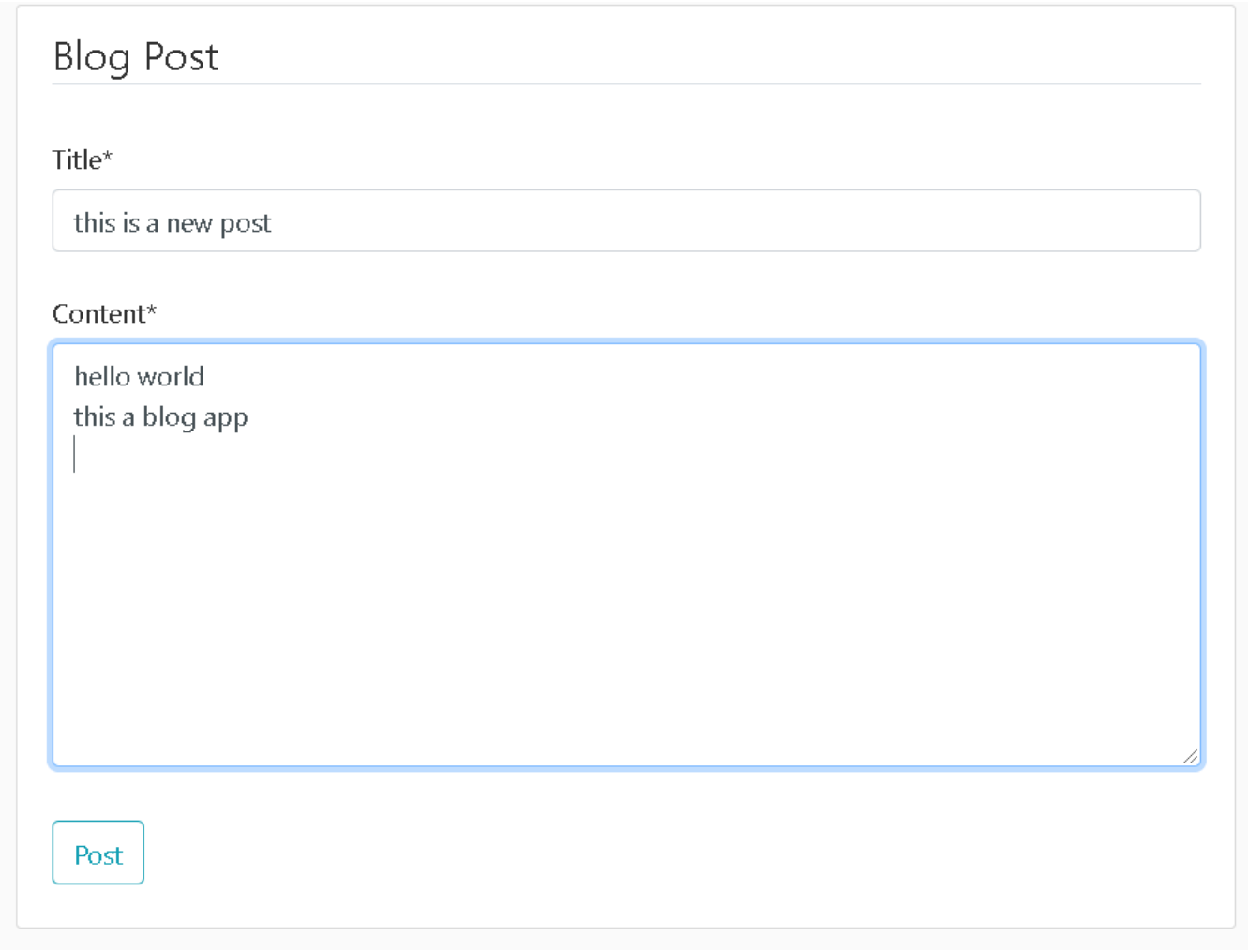
Sign Up

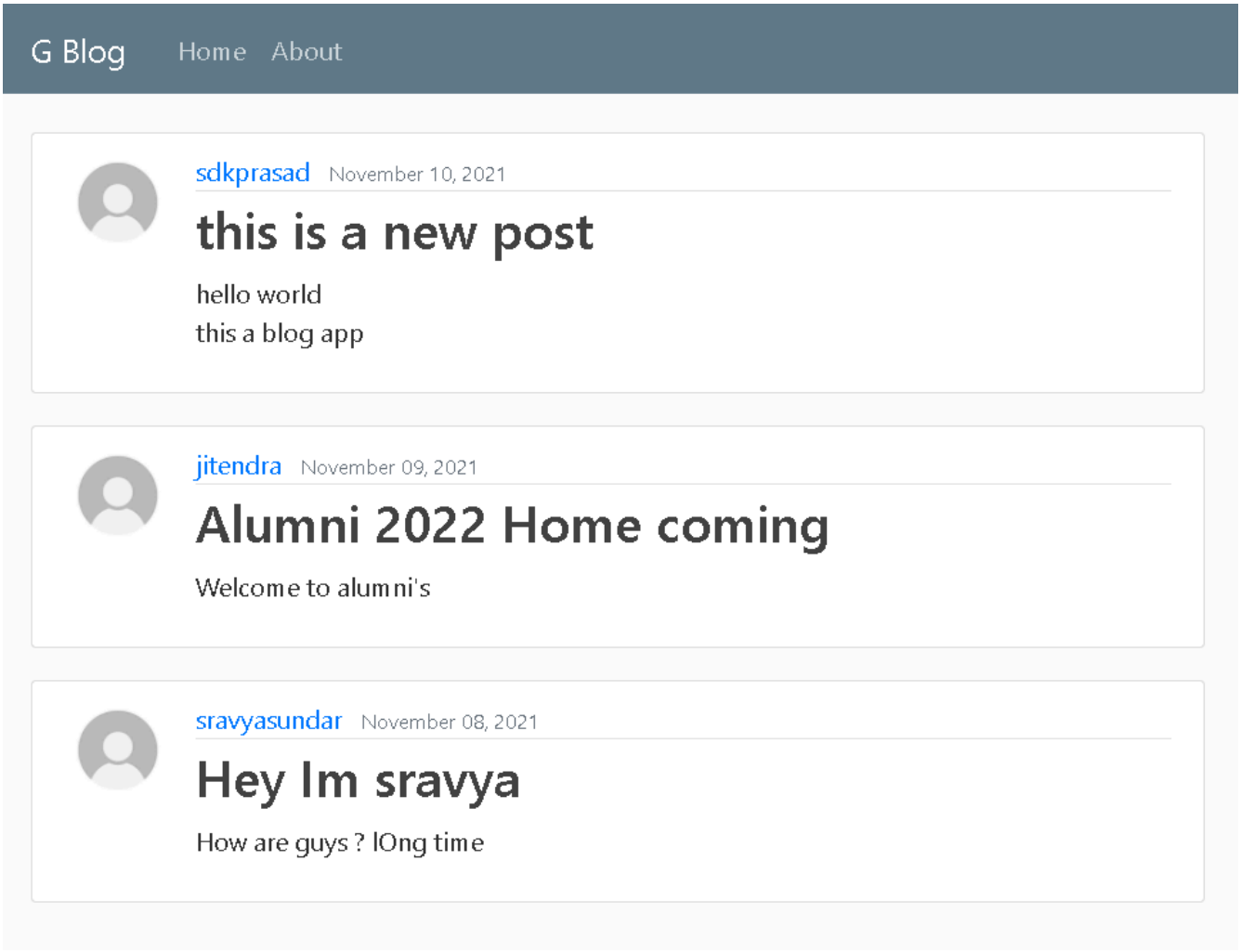


Testcase 8 login page username and password validation

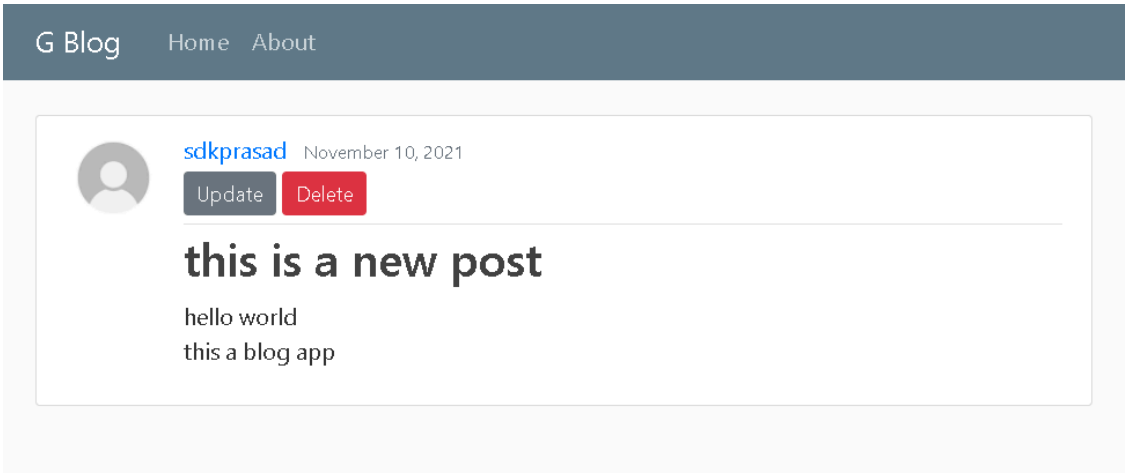


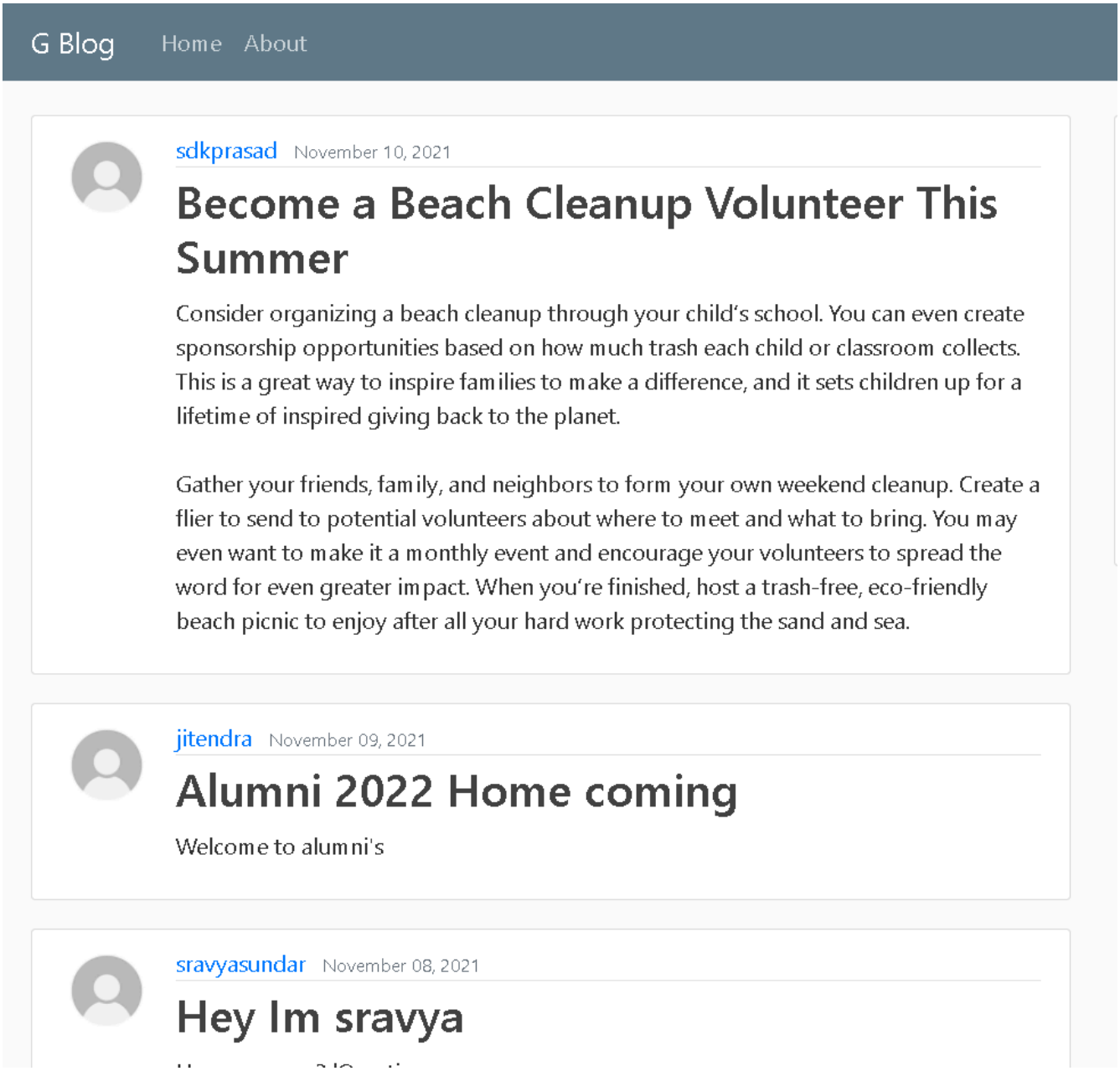
Testcase 9 posting a blog


Blog Post
Title*
this is a new post
Content*
hello world
this a blog app
Post



Testcase 10 updating a blog





Testcase 11: deleting a blog

sdkprasad November 10, 2021

Update Delete

Become a Beach Cleanup Volunteer This Summer

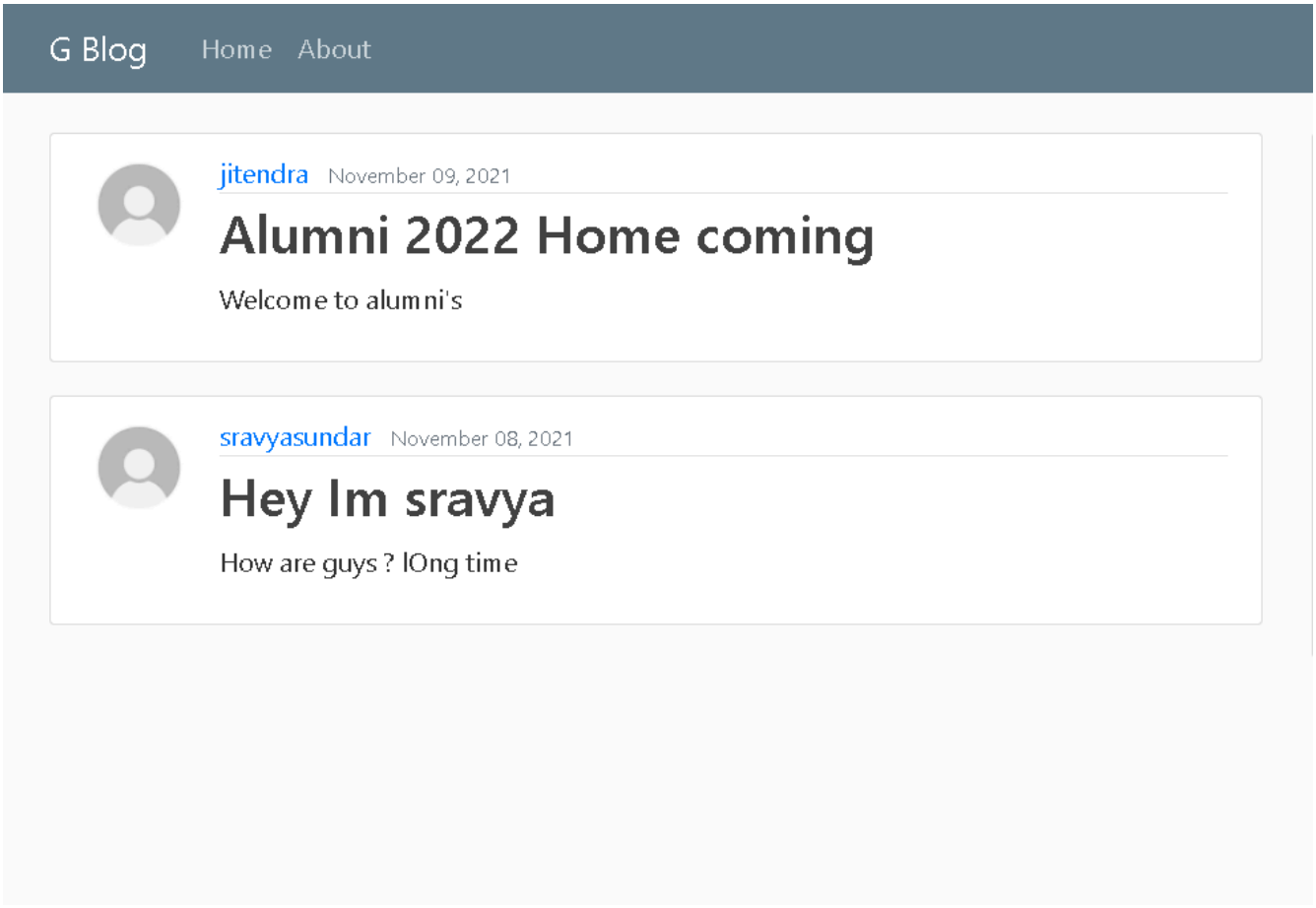
Consider organizing a beach cleanup through your child's school. You can even create sponsorship opportunities based on how much trash each child or classroom collects. This is a great way to inspire families to make a difference, and it sets children up for a lifetime of inspired giving back to the planet.

Gather your friends, family, and neighbors to form your own weekend cleanup. Create a flier to send to potential volunteers about where to meet and what to bring. You may even want to make it a monthly event and encourage your volunteers to spread the word for even greater impact. When you're finished, host a trash-free, eco-friendly beach picnic to enjoy after all your hard work protecting the sand and sea.

Delete Post

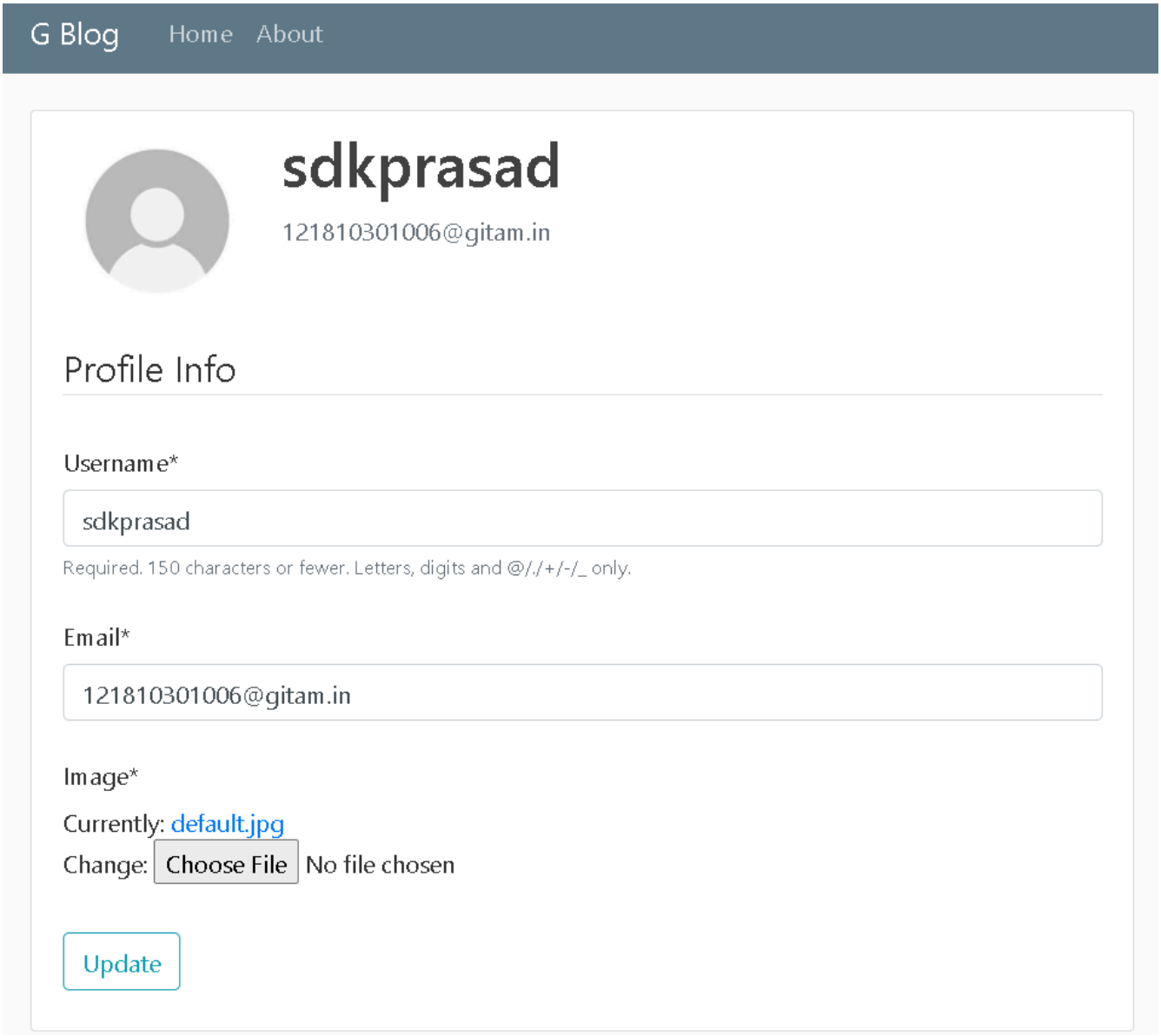
Are you sure you want to delete the post "Become a Beach Cleanup Volunteer This Summer"

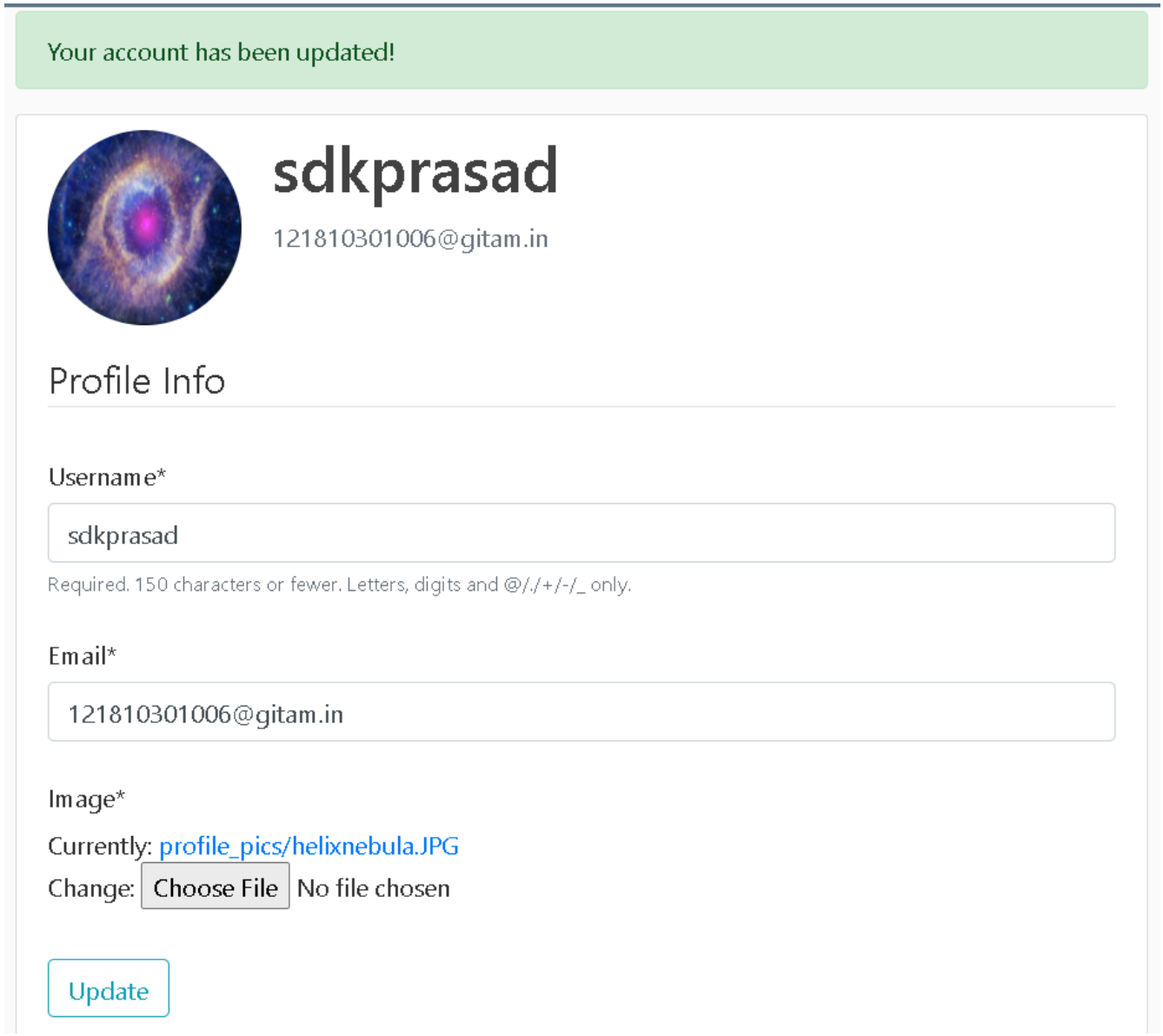
Yes, Delete Cancel



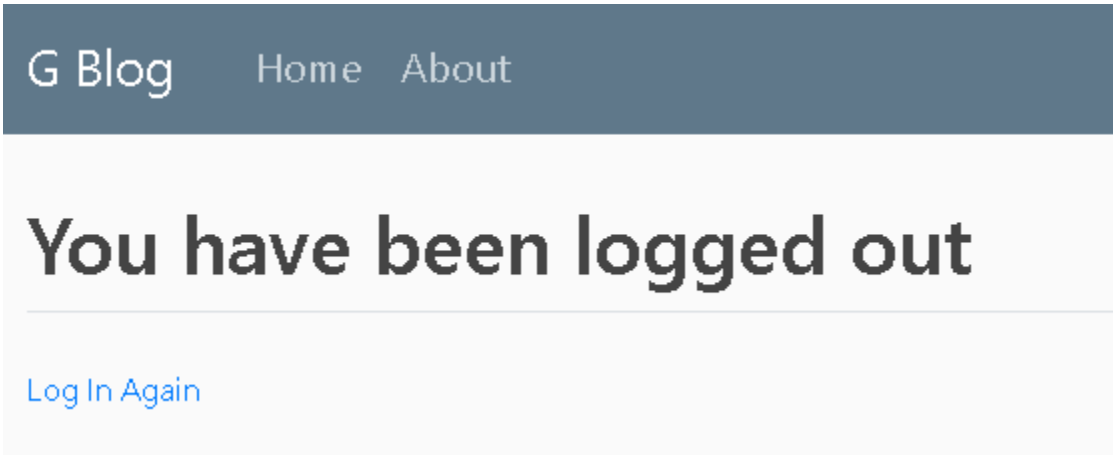
Blog deleted

Testcase 12 changing profile pic

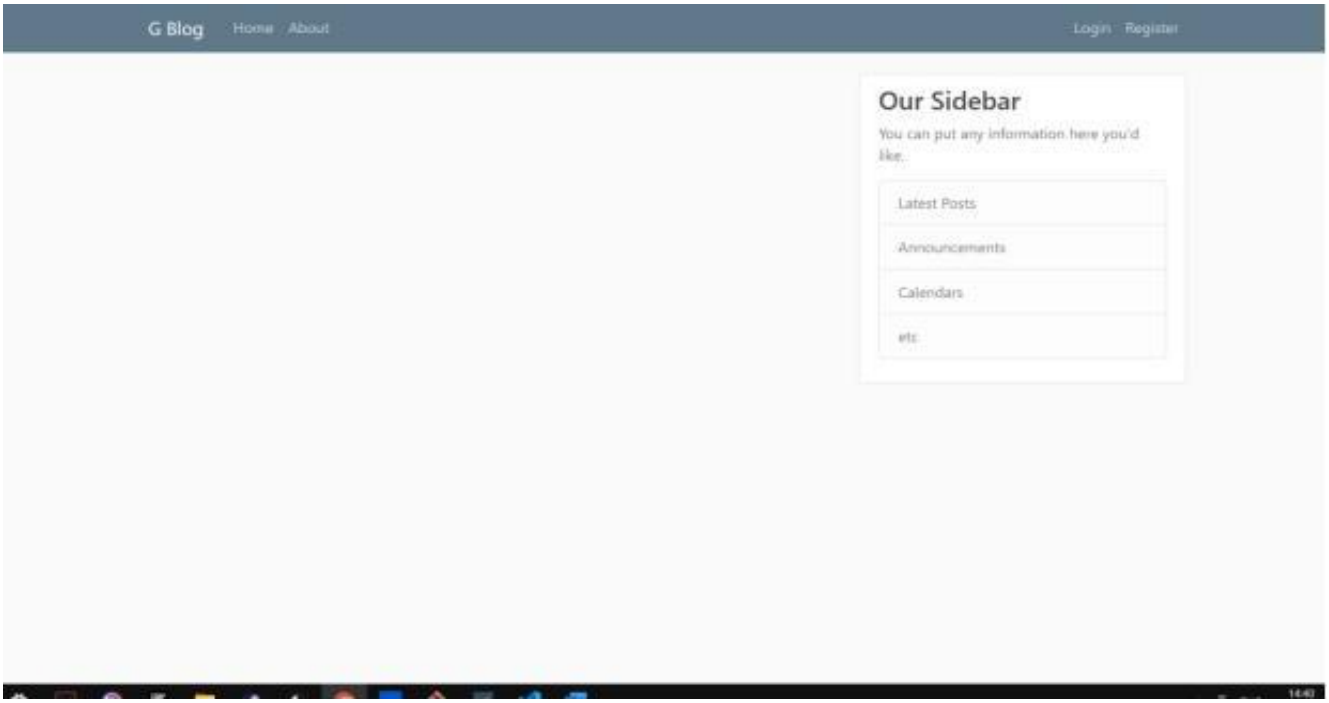

G Blog
Home
About
sdkprasad
121810301006@gitam.in
Profile Info
Username*
sdkprasad
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.
Email*
121810301006@gitam.in
Image*
Currently: default.jpg
Change: Choose File No file chosen
Update

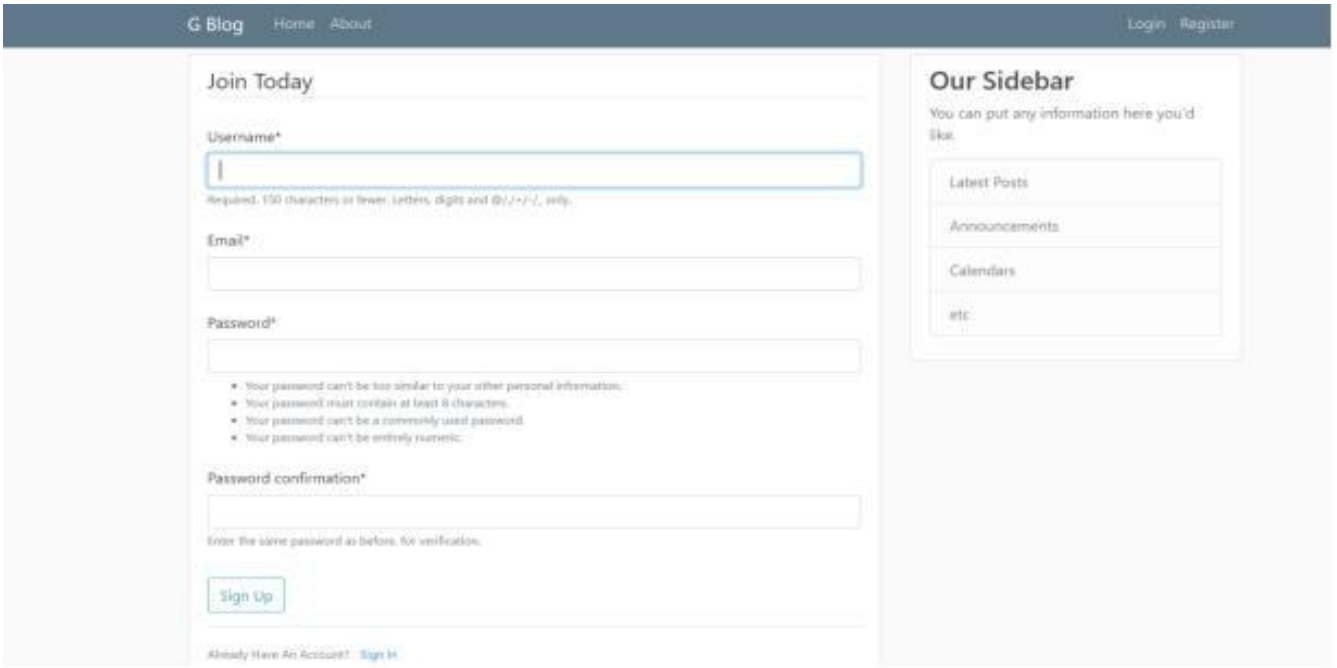


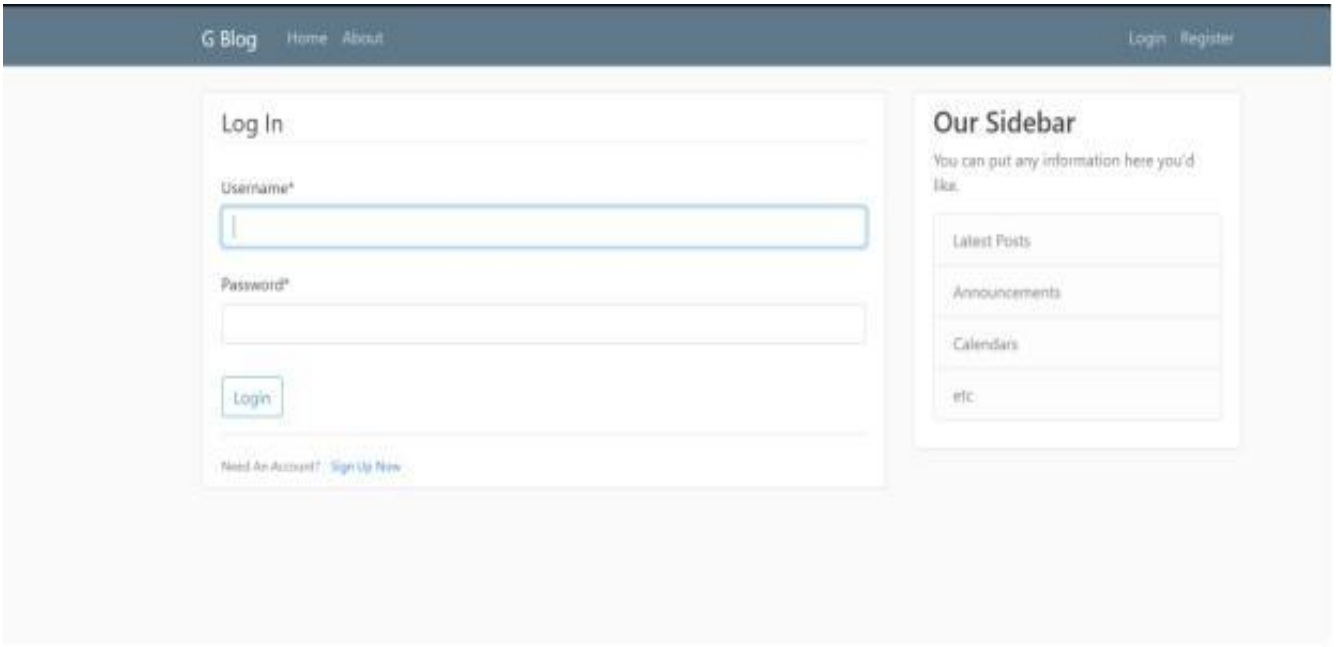
Test case 13 Logging out


G Blog
Home
About
You have been logged out
Log In Again

Results


G Blog
Home
About
Login
Register
Our Sidebar
You can put any information here you'd like.
Latest Posts
Announcements
Calendars
etc


G Blog
Home
About
Login
Register
Join Today
Username*
Email*
Password*
Password confirmation*
Sign Up
Already Have An Account?
Sign In
Our Sidebar
You can put any information here you'd like.
Latest Posts
Announcements
Calendars
etc


G Blog
Home
About
Login
Register
Log In
Username*
Password*
Login
Need An Account? Sign Up Now
Our Sidebar
You can put any information here you'd like.
Latest Posts
Announcements
Calendars
etc

Conclusion & Future Scope

Many elements of our life have been influenced by technology, and research is no exception. Researchers can obtain empirical data on a wide range of issues via blogs, which are a fresh and valuable resource. The widespread use and acceptance of blogs can only increase their usefulness as a research tool.

As you may be aware, the number of blogs is steadily increasing. As a result, blogs with high-quality content survive, while blogs with lower-quality content fail to succeed on the internet. The younger age is shifting away from static content and toward dynamic content such as flash and real-time information such as Twitter.In social networking sites, Twitter is quite valuable. The majority of bloggers use Twitter to promote their websites. So, if you want to stay up to date on the latest blog news, you should follow Twitter. Nowadays, you don't have to worry about how to receive the most up-to-date information; all you have to do is go to the website of your choice and look at the home page. The website's main page provides you with the most up-to-date information on the site's content. Our website, which is only for gitamites, is a blogging platform that can quickly become famous among students and ex-students. As fresh students enrol at the university, the number of users will grow year after year.

References

https://docs.djangoproject.com/en/2.1/

https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction