University Of Information Technology And Sciences (UITS)

SRS Document

Project Title:
— Minimal, Distraction-Free Blogging Web App

Course:
Software Engineering and System Analysis Lab
CSE356

Submitted By:
Zobayer Hasan
ID: 2215151106
Md. Tanveer Ahammed Tarek
ID: 2215151109
Mohammad Shakibul Hasan
ID:2215151115
CSE 51 (6C1)

Submitted To:
Mr. Md Safaet Hossain,
Associate Professor,
Department of CSE,
University Of Information Technology And Sciences

Propa Punam
Lecturer,
Department of CSE,
University Of Information Technology And Sciences

Table of Contents: **Page no:**

1. Introduction

1.1 Purpose

The purpose of this document is to outline the software requirements for the development of a Minimal, Distraction-Free Blogging Web App. The focus of this project is to deliver a user-friendly, minimalistic blogging platform that emphasizes simplicity and distraction-free content creation and management.

1.2 Document Conventions:

1.3 Intended Audience

Users who likes clean, minimal and destraction-free environment.

1.4 Project Scope

The Minimal, Distraction-Free Blogging Web App will offer users a simple platform to create, manage, and publish blog posts without distractions such as ads or cluttered interfaces. It will focus on a clean user experience with essential features for blogging, such as a rich text editor and markdown support, as well as customizable themes for day and night modes.

1.5 References

Minimal Design:
https://www.canva.com/learn/minimalist-design-beautiful-examples-and-practical-tips/
Ted Blog: https://blog.ted.com/

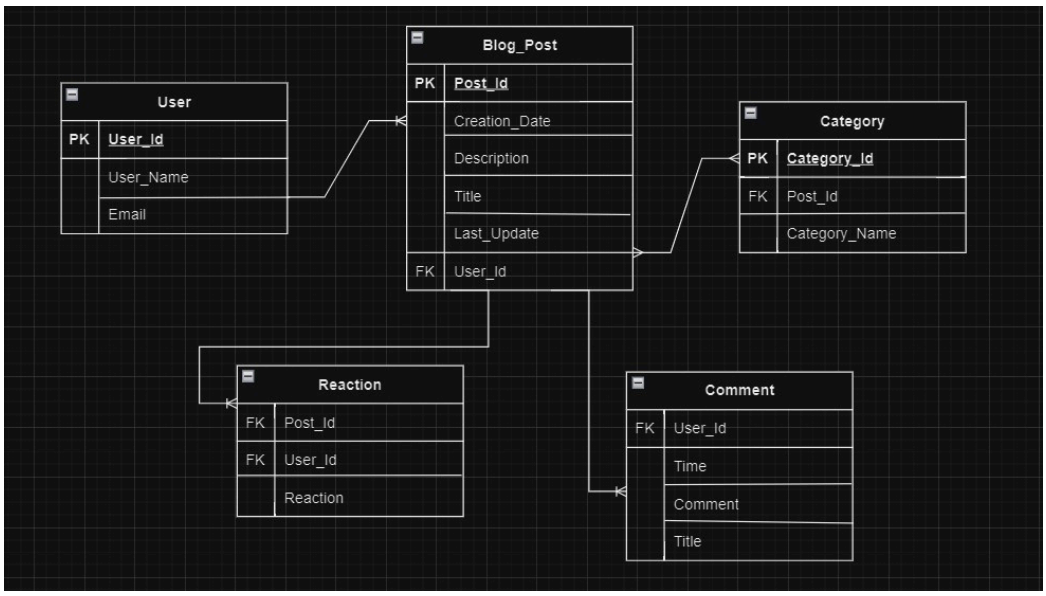
2. Overall Description

2.1 Product Perspective

The product is a web-based blogging platform that eliminates unnecessary features found in many existing platforms, focusing purely on content creation and management. Built using the Django framework, the app will ensure fast load times, a smooth user experience, and reliable security for user data. It offers a minimalistic design and distraction-free mode to improve writing focus and user engagement.

2.2 Product Features

The major features of the bloging database system as shown in below entity–relationship model (ER model)



2.3 User Classes and Characteristics

Blog Writers: The primary users who will write and publish blog content.
Blog Readers: Users who will read content published on the platform.
Admin Users: Responsible for overseeing content, managing users, and handling backend maintenance tasks.

2.4 Operating Environment

1.Web Browsers:

- Fully compatible with modern browsers, including:
- Google Chrome
- Mozilla Firefox
- Apple Safari

2. Server Environment

- Developed and hosted using Django's built-in server for local testing.
- Production deployment is intended for platforms like Heroku or Vercel.

3. Database Environment:

- **Local development:** SQLite for simplicity and ease of configuration.
- **Production:** Scalable PostgreSQL or other relational databases.

4. Supported Devices

- Desktops and laptops with standard browser capabilities.
- Responsive design ensures compatibility with tablets and mobile devices.

2.5 Design and Implementation Constraints

1. Technology Constraints

- Development is centered around the Django framework, ensuring compatibility with Python-based libraries.
- Initial database implementation is limited to SQLite, with provisions for migration to PostgreSQL for scalability.

2. Time Constraints

- The project must be completed within a semester, requiring phased development and strict adherence to deadlines.

3. Resource Constraints

- Limited hardware resources during development, necessitating efficient use of local environments.
- Small team size, which imposes restrictions on extensive testing or feature expansion within the timeframe.

4. User Interface Design Constraints

- The interface must follow minimalist design principles to align with the project's vision.

2.6 Assumptions and Dependencies

1. Assumptions

- Users have basic internet access and knowledge of using web applications.
- Minimal downtime will occur during deployment and updates.
- Day and night mode customization will cater to the majority of users' preferences.

2. Dependencies

Framework and Libraries:

- Reliance on the Django framework and associated Python libraries for core functionality.

Hosting Services:

- Dependence on platforms like Heroku or Vercel for scalable and reliable hosting.

Third-Party Tools:

- Markdown rendering libraries for editor functionality.
- Secure authentication tools for user login and data encryption.

3. System Features

3.1 Functional Requirements

- User Registration and Authentication: Users should be able to create accounts, log in, and manage their profiles.
- Blog Post Creation and Management: Users can create, edit, and delete blog posts using a simple, distraction-free interface.
- Rich Text and Markdown Editor: Provide support for both rich text and markdown for users to format their posts.
- Customization Options: Day/night themes and font adjustments should be available to customize the user reading and writing experience.
- Minimalistic UI: A clean interface that eliminates unnecessary elements, ensuring focus on content creation and consumption.
- Content Publishing: Users can publish their posts, which will be available publicly or privately, depending on their preferences.

4. External Interface Requirements

4.1 USER INTERFACES

The User Interfaces for the Minimal, Distraction-Free Blogging Web App will adhere to the principles of minimalism and usability.

1. Login and Registration Pages

- Clean and simple forms with input validation for user credentials.
- Prominent call-to-action buttons for logging in, signing up, and password recovery.

2. Blog Creation and Editing Interface

- Distraction-free editor with:
- Rich text and markdown support.
- Toolbar with essential formatting options (bold, italics, headings, lists).
- Preview mode to view the final layout before publishing.

3.Customization Options

Easily accessible settings for:

- Switching between day/night modes.
- Adjusting font styles and sizes.

4. Reader Interface

Minimalistic layout for blog posts with:

- Pagination or infinite scrolling for smooth navigation.
- Clean typography and a clutter-free reading experience.

5. Admin Dashboard

- Centralized interface for managing user accounts, blog content, and analytics.
- Secure access to features like content moderation and platform settings.

4.2 SOFTWARE INTERFACES

Web Browsers: The app must function on modern browsers, such as Chrome, Firefox, and Safari.
Database: Initially, SQLite will be used for development, with potential migration to PostgreSQL for scaling.
Third-Party Libraries: The Django framework, along with any required libraries for markdown support, UI design, and security, will be integra ted.

4.3 COMMUNICATION INTERFACES

The app will include robust communication protocols to ensure secure and efficient data exchange.

1. HTTP/HTTPS Protocols

- All data transmission between the client and server will use secure HTTPS to ensure encryption and prevent interception of sensitive information.

2. RESTful API for Backend Communication

- The backend will expose RESTful APIs for core functionalities, including:
- User authentication and profile management.
- Blog content creation, retrieval, and deletion.
- Admin operations for user and content moderation.

3. Web Sockets (Optional)

- For real-time updates (e.g., live collaboration or notifications), WebSocket communication may be implemented in future iterations.

4. Email Communication

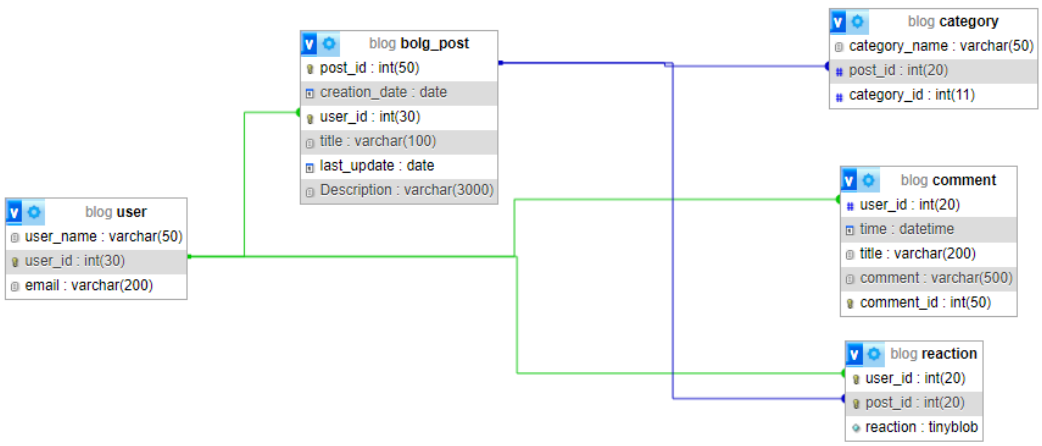
- Integration with SMTP or email services for:
- Account verification emails.
- Password reset links.

5. Non-functional Requirements

5.1 Performance Requirements

- **Load Times:** The web app must have a fast response time, with pages loading within 3 seconds.
- **Scalability:** The app should be able to handle increasing traffic as the user base grows, with options for scaling on cloud platforms like Heroku or Vercel.

Database Relationship:



5.2 Safety/Security Requirements

- **Data Encryption:** All sensitive data (e.g., passwords, user information) must be securely encrypted both at rest and in transit.
- **Backup Procedures:** Automated backups will be implemented to prevent data loss.
- **User Authentication:** Secure user authentication processes, including password hashing, will be employed to safeguard accounts.

5.3 Software Quality Attributes

1. Usability
Ease of Use: The platform's minimalistic and intuitive design ensures that both writers and readers can navigate and use the features without prior training.
Accessibility: Responsive design ensures compatibility with various devices and screen sizes, including desktops, tablets, and mobile phones.

2. Performance Efficiency
Load Times: Pages must load within 3 seconds to maintain user satisfaction and avoid drop-offs.
Resource Utilization: Efficient use of server and database resources ensures the app performs well even with limited hosting environments.

3. Reliability
Fault Tolerance: The app should gracefully handle unexpected errors, such as temporary server unavailability or user input errors, without data loss.
Data Integrity: Automated backups and secure data handling ensure user content remains safe.

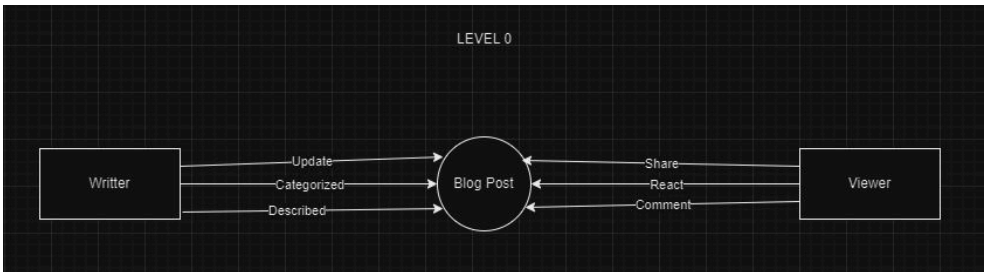
4. Security
User Authentication: Robust password hashing and secure login mechanisms protect user accounts.
Data Encryption: All sensitive information is encrypted both at rest and in transit.
Authorization Control: Role-based access ensures that only authorized users (e.g., admins) can access specific features.

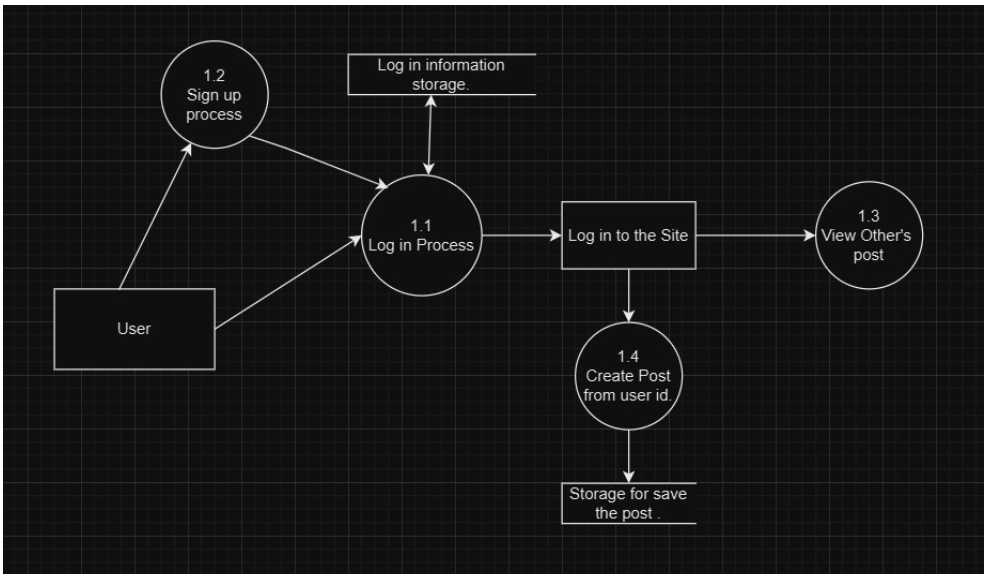
5. Maintainability
Code Modularity: Clear separation of concerns within the codebase simplifies updates and bug fixes.
Documentation: Comprehensive developer documentation allows for easier onboarding of new team members and future scalability.

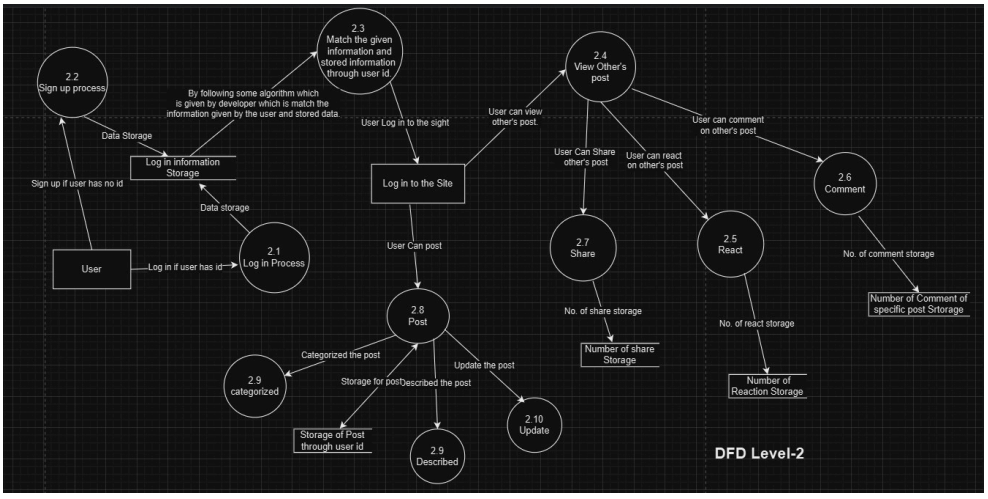
6. Scalability
Horizontal Scalability: Designed to handle increasing user traffic by scaling across additional servers or cloud resources.
Vertical Scalability: Efficiently supports growing demands on databases and storage with options for upgrades.
7. Portability
Deployment Flexibility: The app can be deployed on various hosting platforms like Heroku, Vercel, or self-hosted servers with minimal configuration.
Browser Compatibility: Operable on all modern browsers (Chrome, Firefox, Safari).

6.UML

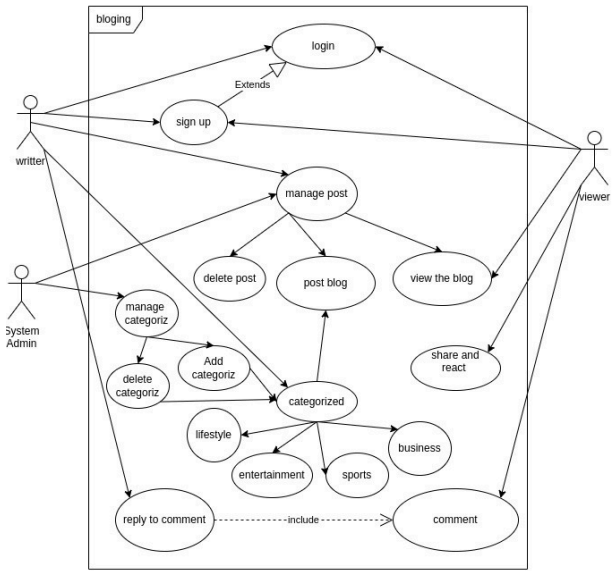
DFD:



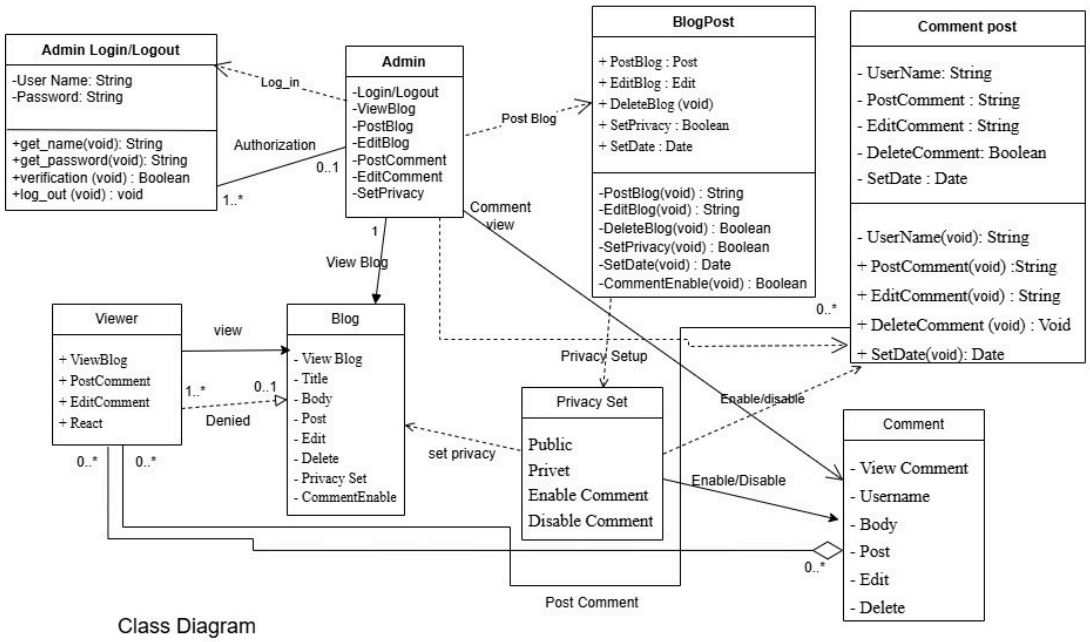




use case:



Class diagram:



Class Diagram

Activity Diagram:

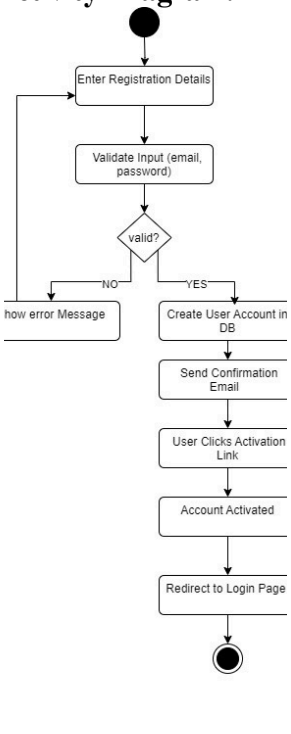


fig: User Registration Activity Diagram

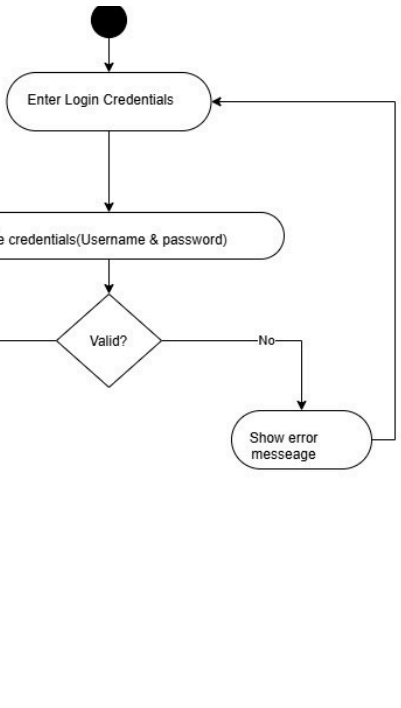
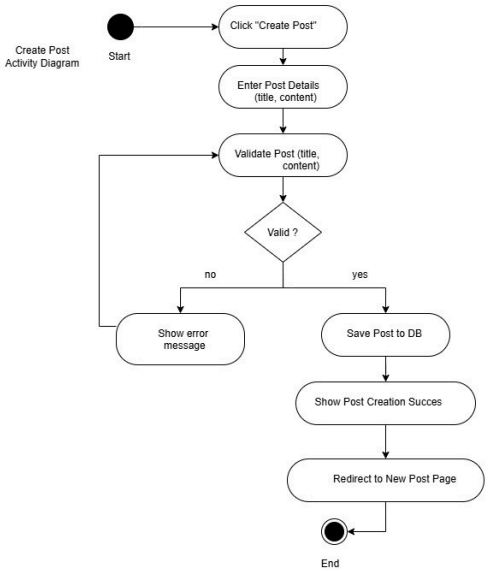
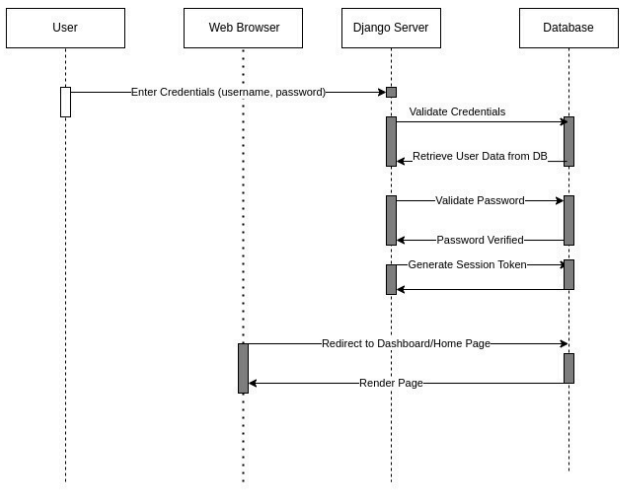


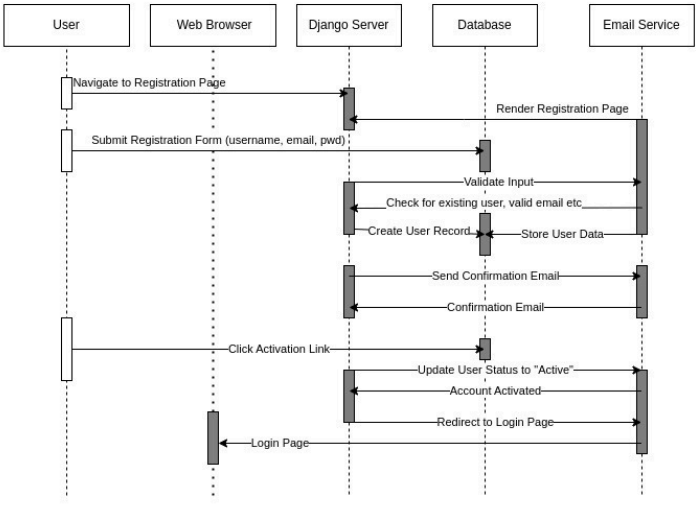
Figure: User Longin Activity Diagram



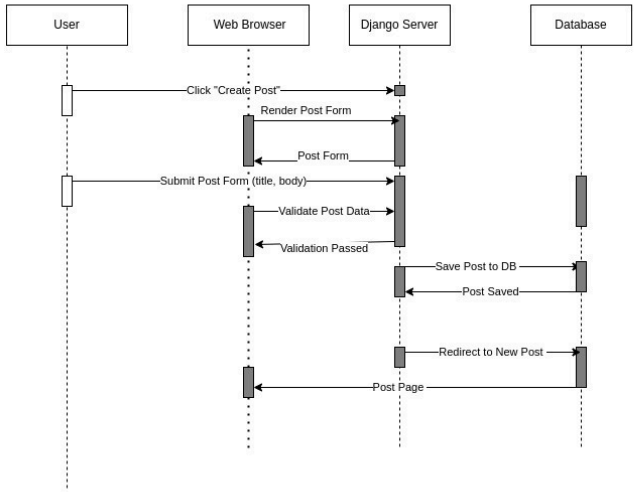
Sequence diagram



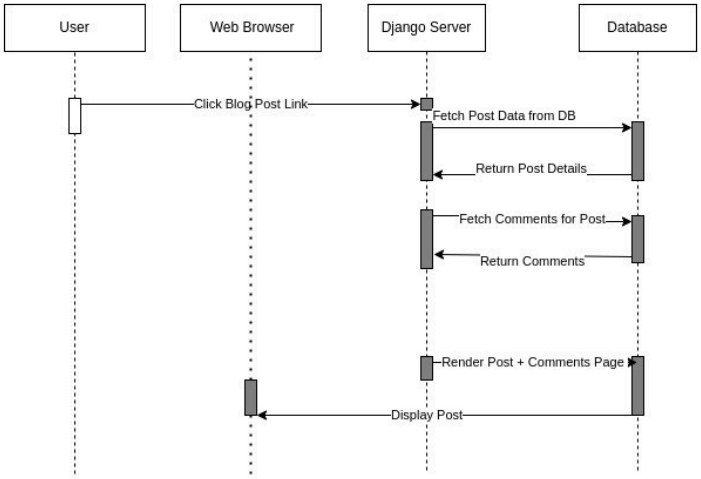
User Login Sequence Diagram



User Registration Sequence Diagram



Creating a Blog Post Sequence Diagram



Viewing a Blog Post Sequence Diagram

gantt and Budget Chart:

gantt chart:

name	status	startDa	dueDat
Complete project Execution	Done	8/22	12/8
Requirement	Done	8/23	8/31
System Analysis	Done	9/1	9/12
Modelling	Done	9/19	11/27
ER Diagram	Done	9/19	9/26
Data Flow Diagram	Done	9/26	10/3
Use case Diagram	Done	10/10	10/17
Class Diagram	Done	10/24	10/31
Activity Diagram	Done	11/2	11/7
Sequence Diagram	Done	11/7	11/14
Gantt & Budget Chart	Done	11/14	11/27
Impliment	Done	9/25	12/5
Creating django backend	Done	9/6	10/4
Index page and home page	Done	9/20	10/10
Log in and Signup	Done	10/11	11/10
Create post	Done	10/25	11/22
Created share	Done	11/15	11/24
Created comment	Done	11/24	11/30
Created like	Doing	11/30	12/5
Testing	todo	11/20	12/8

Budget chart

Estimated Budget for Django Based Online Blogging Website.			
Category	**Description**	**Estimated Cost**	**Notes**
1. Development Tools	Tools and software for coding (IDEs, text editors)	$100	Budget for paid tools if needed.
2. Web Hosting	Cost of hosting the blog (AWS, Heroku, DigitalOcean)	$200	Monthly cost of hosting.
3. Domain Name	Purchase of the domain name (e.g., example.com)	$20	Annual cost of the domain.
4. Design & UI/UX Tools	Subscription to design tools (Canva Pro)	$50	If using premium design tools.
5. Third-party APIs/Services	Subscription to third-party APIs (Google Maps, Mailchimp)	$60	For blog features like email subscription.
6. Development Team Labor	Developer salaries or hourly wages (if outsourced)	$3,500	Hourly rate or fixed cost of developers.
7. Testing and QA	Cost of testing services or tools (paid testing software)	$100	If using professional testers or tools.
8. Marketing and Promotion	Budget for digital marketing (ads, SEO tools)	$150	For online advertising.
9. Miscellaneous	Any unforeseen expenses (additional resources, extra licenses)	$50	Unplanned costs.
10. Contingency Fund	Set aside for unforeseen issues and emergencies	$200	Reserve funds for project issues.
Total Estimated Budget:		**$4,430**	

Task Set

Project Type:

Web Application Development Project that is applicable when Internet-based web application and web site is developed for online blogging.

Adaptation Criteria:

Size of project:

Between 1 and 3 person for 3 months and/or 10 to 15 function points.

Number of potential users:

To be used by local staff only, no immediate outside use.

Mission criticality:

R&D project, no immediate business impact.

Application longevity:

Use projected to be over 7 years.

Stability of requirements from external customer:

The scope is documented and well understood.

Ease of customer-development communication:

User is the developer. They can develop the site externally by posting blog.

Maturity of technology:

Technology is well understood and mature.

Performance constraints:

Run-time performance and/or data capacity are important.

Project staffing:

Internal staff, high competency and experience.

Interoperability:

The application is standalone but does access an external file structure or database.

Reengineering factors:

The system is to be reengineered with minor changes to requirements; descriptive information must be reworked.

8. Conclusion

The project aims to provide a streamlined blogging experience for writers and readers with minimal design and essential features. It prioritizes usability, performance, and security, aligning with the modern needs of content creators. While constrained by time and resources, the modular design allows for future scalability and integration with advanced tools or third-party services.