EDGE PROJECT: Department of ICT, MBSTU

Deliverable

Project title: Personal Poetry and Connection Platform

Submitted by:Mst.Mostary Khatun

Supervisor Name: Dr.Ziaur Rohman

September 19, 2024

Date	Revision	Release Notes
11-09-2014	Rev 01	Initial Release

Contents

1 Abstract **2**

2 Introduction **2**

3 Completion Plan **2**
3.1 Phase 1: Planning and Requirements Gathering 2
3.2 Phase 2: Design 3
3.3 Phase 3: Development 3
3.4 Phase 4: Testing 3
3.5 Phase 5: Deployment 3
3.6 Phase 6: Maintenance 3

4 Proposed Application Details **4**
4.1 Features: 4
4.2 Technical Stack: 4

5 Discussion **4**
5.1 Challenges: 4
5.2 Benefits: 5
5.3 Future Enhancements: 5

6 Conclusion **5**

1 Abstract

The Personal Poetry and Connection Platform is a web-based application built using Django that enables users to create, share, and connect through poetry. The platform allows individuals to write and store their own poems, explore works by others, and engage in discussions around various poetic themes. Key features include a user-friendly interface for poem submission, tagging, and categorization by emotions or topics, as well as a system for user interaction and feedback. The platform aims to foster a community where poetry enthusiasts can connect and collaborate.

2 Introduction

The Personal Poetry and Connection Platform is designed to address the need for a creative and interactive space dedicated to poetry. While many platforms exist for general social networking, few focus solely on poetry and its appreciation. This project leverages Django's robust framework to provide a feature-rich environment for users to:

- Create and manage personal poetry collections.
- Discover and engage with poems from other users.
- Tag and categorize poems by emotions or themes.
- Participate in discussions and provide feedback on poetic works.

The platform is intended to cater to poets, poetry enthusiasts, and anyone interested in exploring the art of poetry.

3 Completion Plan

3.1 Phase 1: Planning and Requirements Gathering

- Define Objectives: Outline key features and functionalities.
- User Stories: Create user personas and use cases.
- Technology Stack: Finalize technologies and tools, including Django, PostgreSQL (or SQLite), and front-end technologies.

3.2 Phase 2: Design

- UI/UX Design: Design wireframes and mockups for user interface.
- Database Design: Develop schema for poems, users, tags, and comments.
- Architecture: Define system architecture including Django app structure.

3.3 Phase 3: Development

- Set Up Django Project: Initialize Django project and configure settings.
- Develop Models: Create models for Users, Poems, Tags, and Comments.
- Build Views and Templates: Implement views and templates for poem management, user interactions, and browsing.
- Integrate User Authentication: Add user registration, login, and profile management.
- Implement Tagging and Categorization: Develop functionality for tagging poems and filtering by emotion or theme.

3.4 Phase 4: Testing

- Unit Testing: Write and execute unit tests for models and views.
- User Testing: Conduct user testing sessions to gather feedback and identify issues.
- Bug Fixes: Address any bugs or usability issues identified during testing.

3.5 Phase 5: Deployment

- Deploy Application: Set up the production environment and deploy the application.
- Monitor Performance: Implement monitoring and logging to track application performance and errors.

3.6 Phase 6: Maintenance

- Ongoing Support: Provide support for users and fix issues as they arise.
- Feature Enhancements: Continuously improve the platform based on user feedback.

4 Proposed Application Details

4.1 Features:

- User Registration and Login: Secure user authentication with Django's built-in authentication system.
- Poem Management: Interface for users to add, edit, and delete their poems.
- Poem Discovery: Browse and search for poems based on tags, themes, or user preferences.
- Tagging and Categorization: Apply and filter poems by tags related to emotions or themes.
- User Profiles: Users can view and edit their profiles, and see their own and others' poems.
- Comments and Feedback: Allow users to comment on and provide feedback on poems.
- Responsive Design: Ensure the platform is usable on various devices.

4.2 Technical Stack:

- Backend: Django with Django REST Framework (for APIs, if needed)
- Frontend: HTML, CSS, JavaScript (with frameworks like Bootstrap for styling)
- Database: PostgreSQL (preferred for production) or SQLite (for development)
- Hosting: Platforms like Heroku, AWS, or DigitalOcean for deployment

5 Discussion

5.1 Challenges:

- User Authentication: Ensuring secure user authentication and data protection.
- Performance: Handling large volumes of poems and user interactions efficiently.
- Scalability: Designing the system to handle growth in users and data.

5.2 Benefits:

- Community Building: Fosters a community of poetry enthusiasts and creators.
- Creative Expression: Provides a platform for individuals to express and share their creativity.
- Interactive Features: Engages users through comments and feedback, enhancing user interaction.

5.3 Future Enhancements:

- Advanced Search: Implement more sophisticated search features based on content analysis.
- Mobile App: Develop a companion mobile app for on-the-go access.
- Social Integration: Integrate with social media platforms for sharing and broader reach.

6 Conclusion

The Personal Poetry and Connection Platform aims to provide a dedicated space for poetry enthusiasts to create, share, and engage with poetry. By leveraging Django's robust framework, the application offers a user-friendly and scalable solution to foster a vibrant poetry community. The project's completion will mark a significant step towards enhancing poetic expression and interaction, creating a valuable resource for poets and readers alike.