A

MAJOR PROJECT-III REPORT

on

StoryGram

Submitted by:

Harshit Jain (220557)
Nikhil Kumar (220489)
Aranav Kumar (220485)
Ayush Sengar (220486)

under mentorship of

Dr. Atul Mishra
Associate Professor

Department of Computer Science Engineering
School of Engineering and Technology
BML MUNJAL UNIVERSITY, GURUGRAM (INDIA)

May 2025

CANDIDATE'S DECLARATION

I hereby certify that the work on the project entitled**," Project Name – StoryGram (A Collaborative Story Writing Platform)"**, in partial fulfilment of requirements for the award of Degree of Bachelor of Technology in School of Engineering and Technology at BML Munjal University, having **University Roll No. 220557, 220489, 220485, 220486** is an authentic record of my own work carried out during a period from **January 2025 to May 2025** under the supervision of **Dr. Atul Mishra.**

Harshit Jain – 220557
Nikhil Kumar - 220489
Aranav Kumar - 220485
Ayush Sengar - 220486

SUPERVISOR'S DECLARATION

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Faculty Supervisor Name: **Dr. Atul Mishra**

Signature:

ABSTRACT

In an age where storytelling is evolving beyond traditional books and blogs, there remains a gap for collaborative, interactive platforms where multiple voices can shape a narrative together. **Storygram** aims to fill that gap by introducing a fresh approach to digital storytelling - one that blends creativity, community, and control.

Storygram is a collaborative story-writing platform that lets users not only read and write stories, but also contribute directly through a unique pull request model, much like how developers contribute to open-source projects. Instead of leaving simple suggestions, users can write their own additions to a story, submit them as pull requests, and let the original author decide whether to merge them. This gives writers full control over their stories while inviting creative input from the community.

To protect the creative process and maintain intrigue, stories are initially blurred or show just a small preview. Users must accept the platform's terms before gaining access or contributing, ensuring respectful collaboration.

Beyond writing, **Storygram** is also about community. Users can engage through comments, voting, reviews, and build their own profile and reputation. A powerful search and discovery system helps readers find stories that match their taste, while authors benefit from increased visibility and reader feedback.

In short, **Storygram** isn't just a place to write stories - it's a place to co-create, interact, and build a story-driven community, one chapter at a time.

.

ACKNOWLEDGEMENT

I am highly grateful to **Dr. Atul Mishra**, Professor, BML Munjal University, Gurugram, for providing supervision to carry out the Project 3 study from **January-May 2025**.

Dr. Atul Mishra has provided great help in carrying out my work and is acknowledged with reverential thanks. Without wise counsel and able guidance, it would have been impossible to complete the training in this manner.

I would like to express thanks profusely to thank Dr. Atul Mishra, for stimulating me from time to time. I would also like to thank the entire team at BML Munjal University. I would also thank my friends who devoted their valuable time and helped me in all possible ways toward successful completion.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Full Form
AWS	Amazon Web Services
EKS	Elastic Kubernetes Services
ELB	Elastic Load Balancer

Chapter 1

Introduction

Storytelling has always been one of humanity's most fundamental ways of communicating, sharing knowledge, and connecting emotionally. Whether through mythology, folklore, novels, or modern digital platforms, stories have played a crucial role in shaping cultures and societies. In the app era, while publishing and reading stories has become more accessible than ever, storytelling remains largely one-sided. Writers publish, and readers consume. Some platforms offer both reading and writing, but few enable genuine, meaningful collaboration between creators and their audience.

Storygram is an innovative storytelling app designed to redefine the experience by making it interactive, collaborative, and community-driven. It's not just a place to read and write stories-it's a space where readers become co-creators, and writers can engage with their audience on a deeper level. The standout feature of Storygram is its pull request model, inspired by version control systems used in software development. Rather than leaving comments, readers can propose new sections or story continuations. Authors then review and choose whether to merge these into the main storyline. This approach brings structure, transparency, and mutual respect to the creative process.

The app also places strong emphasis on community and engagement. Features like story voting, comment threads, user reviews, and personalized profiles help create a vibrant ecosystem of writers and readers. Built-in discovery tools let users explore content by genre, tags, popularity, or author, ensuring everyone can find stories that resonate. Recognition is built into the platform as well-users gain reputation through accepted contributions, thoughtful reviews, and upvotes from the community.

In essence, Storygram isn't just another storytelling app-it's a creative ecosystem. By combining structured collaboration with the emotional power of narrative, it offers a new way to experience and create stories. With Storygram, we're ushering in a new era of storytelling: open, participatory, and truly engaging.

Chapter 2

Introduction to Project

2.1 Overview

The digital age has transformed the way we create and consume content, but when it comes to storytelling, most platforms still treat it as a linear, one-way process: authors write, and readers simply read. While platforms like blogs, Wattpad, and fanfiction websites provide space for individual writers, there's a clear lack of collaborative, controlled, and community-centered storytelling environments.

Storygram is a collaborative storytelling platform designed to make story creation a shared experience between authors and readers. The platform introduces an innovative pull request model, similar to how developers contribute to code. Readers who wish to add to a story can write a continuation or a new part and submit it as a "story pull request." The original author can then review the contribution and decide whether to accept it, giving them full creative control while still embracing collaborative input.

To maintain the integrity of content and create a more immersive environment, Storygram includes access control features. Stories are blurred by default, and only a short teaser (3-4 lines) is shown initially. Readers must accept terms and conditions before engaging with or contributing to the content. This ensures that authors retain ownership and that only interested, respectful users gain deeper access.

Beyond writing and reading, the platform emphasizes community engagement. Users can interact through comments, votes, reviews, and build their reputation via contributions. Storygram also includes a search and discovery system, allowing stories to be filtered by genre, tags, popularity, and authorship-making it easier for users to find content they love.

In summary, Storygram is not just a platform for writing stories-it's a dynamic, creative ecosystem where stories grow organically with input from a passionate community. It empowers both writers and readers, encourages collaboration, and brings storytelling into a more interactive, thoughtful digital age.

2.2 Existing System

In the current digital storytelling landscape, platforms such as **Wattpad, Medium, Archive of Our Own**, and audio-focused platforms like **KukuFM** have enabled millions of users to share and consume stories. These platforms typically allow readers to follow stories, post comments, and sometimes support authors via likes or donations. They have made storytelling more accessible, especially for amateur and independent writers.

However, these systems are fundamentally author-centric and linear. Stories are usually written and published by a single author or a tightly controlled group, with little opportunity for structured or scalable community involvement. While informal collaboration might occur

through shared prompts or community challenges, there is no dedicated framework for multi-user story development, such as version control or selective contribution acceptance.

For example:

KukuFM focuses primarily on audio stories and podcasts, catering to passive listeners. While it excels in content delivery, it offers no collaborative writing mechanisms or structured story development workflows.

Wattpad allows users to publish serialized fiction and gain feedback, but it does not support pull-request-style contributions or dynamic story branching.

Medium serves more as a blogging platform, with minimal interactive features and no collaborative writing-support.

Additionally, existing platforms offer limited author control over who can contribute or even view content. Once a story is published, it is fully visible to everyone unless manually restricted, and there is no option for blurring previews or requiring reader approval before granting access.

These gaps highlight a missed opportunity to bring the pull request and version control concepts-common in software development-to the creative writing world. Imagine a platform where readers could submit their own story branches, request merges, and interact with authors like collaborators, not just consumers.

Feature	StoryGram	Wattpad	Medium
Collaborative Writing	Yes (Pull Requests)	No	No
Content Access Control	Yes (Blurred Previews)	No	No
Community Engagement	Yes (Voting, Comments)	Yes (Comments)	Yes (Claps)
Primary Focus	Collaborative Stories	Story Publishing	Article Publishing
Version Control	Yes	No	No

Table 1 Comparison of StoryGram with Existing Systems.

2.3 User Requirement Analysis

To develop a platform that effectively serves both writers and readers, a clear understanding of the distinct user roles and their requirements is essential. Authors require the ability to initiate a story and maintain full control over its visibility, ensuring they can manage who gets access to the complete content. They must have the authority to review and approve or reject contributions (akin to pull requests) from readers, thus preserving the story's integrity and direction. Additionally, features like version tracking, contribution history, and credit attribution are crucial for fostering a sense of ownership and collaboration. Reader feedback and reviews also serve as valuable inputs for authors to refine their narratives.

On the other hand, readers and contributors expect a seamless and engaging experience. They should be able to preview stories-either blurred or partially revealed-before fully committing to read or contribute. Before engagement, they must also clearly acknowledge the terms of use.

These users should be empowered to suggest edits or extensions to the story through pull requests, and interact socially through commenting, liking, and voting mechanisms. Discoverability is also a key expectation, requiring robust filtering, trending tags, and search capabilities.

From a platform-wide perspective, the application must support secure user authentication, intuitive user profiles, and real-time notifications for story updates and contribution statuses. Additionally, an easy-to-navigate, aesthetically pleasing interface is essential for both reading and writing experiences. This comprehensive user analysis ensures the platform addresses the nuanced needs of both creators and contributors, fostering a dynamic and collaborative storytelling environment.

Feature	StoryGram	Wattpad	Medium
Collaborative Writing	Yes (Pull Requests)	No	No
Content Access Control	Yes (Blurred Previews)	No	No
Community Engagement	Yes (Voting, Comments)	Yes (Comments)	Yes (Claps)
Primary Focus	Collaborative Stories	Story Publishing	Article Publishing
Version Control	Yes	No	No

Table 2 User Requirements for StoryGram

2.4 Feasibility Study

The development and deployment of the collaborative storytelling application are underpinned by a robust and scalable architecture, ensuring technical, operational, and economic feasibility.

Technical Feasibility: The frontend of the application is crafted using Flutter, facilitating a seamless cross-platform experience for both Android and iOS users. Flutter's single codebase approach ensures consistent UI/UX and streamlines maintenance efforts. On the backend, Golang is employed for its efficiency in handling high-concurrency tasks. The Go HTTP server manages all API interactions, encompassing user authentication, pull request management, commenting, and story versioning. For data storage, PostgreSQL is utilized, containerized within Docker to provide a portable, isolated, and consistent environment across development, testing, and production stages.

Deployment is orchestrated through AWS, utilizing EKS to manage containerized applications. EKS offers a managed Kubernetes environment, simplifying the deployment and scaling of applications. Compute resources are provisioned using Amazon EC2, and API requests are efficiently managed via Amazon API Gateway. To automate integration and deployment processes, AWS Code Pipeline is employed, ensuring synchronized updates between the mobile app and backend services. This pipeline integrates with AWS Code Build to build Docker images and deploy them to the EKS cluster, streamlining the continuous integration and deployment workflow

Operational Feasibility: The application is designed for reliability and scalability. AWS's scalable cloud infrastructure allows the platform to handle increasing traffic seamlessly. ELB

distribute incoming requests evenly across multiple EC2 instances, enhancing availability and reliability. The use of Docker for PostgreSQL simplifies database backups, scaling, and overall management, ensuring smooth operations even as data volumes grow.

Economic Feasibility: Cost efficiency is a cornerstone of the application's architecture. Docker's containerization reduces the complexity and costs associated with managing different environments. AWS's pay-as-you-go model ensures that expenses align with actual usage, making it economical during the early stages with variable traffic. Additionally, the AWS Free Tier offers services like Lambda, S3, and SNS at no cost during development, further reducing operational expenses.

Chapter 3

Literature Review

Recent studies have provided a comprehensive understanding of the mechanisms, roles, and outcomes of online collaborative story writing. Montanelli and Ruskov conducted a systematic literature review of online collaborative storytelling platforms, revealing that collaboration typically occurs in two modes-synchronous interaction suited for dialogue-driven narratives and asynchronous workflows for scene-based contributions-and that common participant roles include writer, consultant, and reviewer, while the facilitator role remains notably absent; they also identified drafting, brainstorming, and reviewing as the predominant activities and called for future work on collaborator recruitment and creative guidance [1][2]. In an educational setting, Wilson examined a high school English class's implementation of Google Docs for a blended collaborative writing project, demonstrating that the platform's real-time editing, version tracking, and cloud-based affordances effectively supported both in-class synchronous co-authoring and asynchronous home writing, though she noted challenges in maintaining consistent student engagement and the necessity of clear pedagogical scaffolding [3][4]. Robayo Luna and Hernández Ortiz investigated project-based collaborative writing among Colombian university students learning English as a foreign language, finding that structured group tasks significantly enhanced learners' ability to organize ideas, develop coherent argumentation, and employ appropriate academic discourse conventions [5][6]. Focusing on interactive methods, another study of online co-writing activities in fully virtual ELICOS courses identified peer prompting, shared task delegation, and integrated feedback loops as key strategies that promoted language acquisition and sustained participation despite the absence of face-to-face interaction [7]. Finally, Li and colleagues evaluated the impact of computer-mediated collaborative writing instruction on Chinese EFL learners, reporting marked improvements in writing performance, self-efficacy, and motivation when students engaged with platforms like Tencent Docs under guided, task-oriented frameworks [8]. Collectively, these findings underscore the multifaceted nature of online collaborative story writing, highlighting the critical interplay between platform affordances, structured pedagogical design, and clearly defined participant roles to maximize both creative output and language development.

3.2 Objectives of the Project

The primary objective of this project is to create a platform that facilitates collaborative storytelling in a way that existing platforms like **Wattpad, Medium, and KukuFM** have not yet fully addressed. The project aims to enable a structured collaboration mechanism where writers can contribute to stories in a manner similar to software development workflows, specifically through the use of pull requests. This feature will allow multiple users to suggest

changes or additions to a story, which will be reviewed and approved by the author, ensuring a controlled and organized writing process.

Additionally, the platform will offer authors more control over their content, specifically in terms of access. Currently, many storytelling platforms lack options to control who can view or contribute to the content once it's published. This project will introduce features like blurred previews and gated access, where users will need explicit approval from the author before they can read the full story or contribute. This will protect the author's creative rights and help prevent unauthorized access and contributions.

Another critical objective is to integrate version control within the platform, allowing authors to track different versions of their stories and manage how and when different parts of the story are updated. This version control system will also enable authors to merge contributions from different users, keeping the story's integrity intact while incorporating diverse ideas and narrative elements. Furthermore, the platform will provide a structured and meaningful way for the community to engage with the stories, allowing readers to vote on story segments, offer suggestions, and provide feedback. This community involvement will ensure that the contributions made are thoughtful, valuable, and aligned with the story's vision.

To ensure a seamless experience across devices, the project will be built using Flutter, which allows the platform to run on both Android and iOS without the need for separate development efforts. This will enable users to collaborate, contribute, and interact with stories on the go, providing a flexible and responsive experience. On the backend, the system will be built with Golang, ensuring high performance, while PostgreSQL and Docker will be used for efficient and scalable database management. The project will also leverage AWS services to provide a robust and scalable infrastructure capable of supporting a growing user base.

Additionally, the platform will improve the discoverability of stories by offering advanced search features, allowing users to find stories based on genre, author, community ratings, and more. These features will help new writers gain visibility and help readers discover new and exciting stories. To ensure the integrity of contributions and protect authors' intellectual property, the platform will incorporate secure and transparent contribution processes, including logs and audit trails that track who contributed what and when, ensuring full transparency.
The ultimate goal is to build a platform that not only supports collaborative storytelling but also creates a dynamic community where authors, readers, and contributors can engage in meaningful ways. By integrating features like pull requests, version control, and community-driven feedback, this platform will revolutionize the way stories are written, developed, and shared, providing a unique space for creativity and collaboration.

Chapter 4

Problem Statement

In the modern digital landscape, storytelling has become more accessible than ever. Writers can self-publish through blogs, forums, and online platforms like Wattpad or Medium, and readers have countless stories at their fingertips. However, most of these platforms follow a one-way communication model: the writer publishes content, and the reader passively consumes it. The level of reader involvement is often limited to likes, brief comments, or star ratings. This creates a gap in creativity and interaction-one where storytelling remains largely a solitary act rather than a collaborative experience.

Despite the rise of online communities, there are very few platforms that allow readers to directly and meaningfully contribute to a story. While some tools offer co-authoring features, they usually lack structured workflows to handle multiple contributions, revisions, and approvals. Without a clear system of control and accountability, collaborative storytelling often becomes disorganized, inconsistent, or overwhelming for authors to manage. There’s no way for readers to propose changes in a controlled environment, nor is there a version history to track the evolution of a story.

In addition, content control and author rights are often not prioritized. Most platforms display the entire story to any visitor, which can lead to issues like unauthorized copying or low-quality feedback from uninvested readers. There's also a lack of structured mechanisms to manage reader engagement, moderation, and content access. This not only affects the quality of content shared but also discourages serious writers from participating fully in open communities.

Finally, discoverability and community recognition are often weak. High-quality stories or emerging writers struggle to get noticed in crowded platforms dominated by trending or popular content. There is a need for smarter feedback systems, structured reviews, and voting mechanisms to help surface content based on value rather than just popularity.

Storygram addresses these problems by introducing a new model: a collaborative storytelling platform that blends the control of traditional authorship with the creativity of open contribution. Using a pull request system, it allows readers to submit story continuations, which authors can review, accept, or decline. By integrating blurred content previews, access permissions, community voting, and feedback**, Storygram** systems offers a fresh and secure way to co-create stories-bridging the gap between writers and readers in a thoughtful, engaging, and well-managed environment.

Chapter 5

Methodology

5.1 Introduction to Languages (Front End and Back End)

Our project utilizes a modern and efficient technology stack tailored for building scalable, responsive, and maintainable mobile applications.

Front-End Language Flutter (Dart) is used for front-end development. Flutter enables cross-platform mobile app development using a single codebase for both Android and iOS. Dart, the language behind Flutter, is optimized for UI development, offering hot reload, smooth animations, and a reactive framework.

Back-End Language Golang (or Go) is used for developing the backend APIs and business logic. Go is known for its performance, simplicity, and built-in concurrency model using goroutines, making it ideal for handling simultaneous user interactions and collaborative tasks in real time.

Component	Role
Flutter (Dart)	Front-End (Cross-platform mobile apps, smooth UI)
Golang	Back-End (High-performance APIs, concurrency with goroutines)
PostgreSQL	Database (Robust, portable data storage in Docker)
AWS EKS	Deployment (Scalable Kubernetes clusters)
AWS EC2	Deployment (Compute instances for hosting)
AWS S3	Deployment (Storage for static assets)
AWS API Gateway	Deployment (API management and scaling)

Table 3 Technology Stack Components

5.2 Any other Supporting Languages/ packages

The collaborative storytelling application utilizes PostgreSQL as its relational database, chosen for its robustness, reliability, and advanced features like extensibility and indexing. To ensure portability and consistency across environments, PostgreSQL is containerized using Docker. This setup facilitates the storage of structured data, including user profiles, stories, comments, and version histories, providing a solid foundation for the application's data management needs.

For deployment, the application leverages AWS, specifically utilizing Amazon EKS to orchestrate containerized services. EKS offers a scalable and managed Kubernetes environment, simplifying the deployment and management of applications. Within this infrastructure, services such as Amazon EC2 provide the underlying compute resources, while Amazon S3 is employed for object storage needs, such as storing media assets or backups. To

manage API requests efficiently, Amazon API Gateway is integrated, ensuring reliable and scalable API interactions.

5.3 User characteristics

The collaborative storytelling application is thoughtfully designed to cater to two primary user groups: **Writers (Authors)** and **Readers (Collaborators),** each bringing distinct motivations and expectations to the platform.

Writers (Authors) are creatively driven individuals who seek a platform to craft and publish their narratives. They value the ability to maintain control over their story's direction while being open to collaborative input that enhances their work. These authors desire features that provide feedback mechanisms, version histories, and structured reviews of contributions, ensuring that any collaborative efforts align with their vision. Moreover, they prefer environments where they can dictate who has access to view or modify their content, safeguarding the integrity of their creations.

On the other hand, **Readers (Collaborators)** are active participants who relish the opportunity to engage in storytelling beyond passive consumption. They are eager to contribute creatively, suggesting ideas or additions to existing stories. These collaborators are drawn to platforms that highlight trending, reviewed, or top-rated stories, allowing them to identify and engage with content that resonates with their interests. Recognition for their accepted contributions or constructive feedback is also important to them, as it validates their involvement and encourages continued participation.

5.4 Constraints

While developing and deploying the collaborative storytelling application, several technical, functional, and operational constraints must be considered. These constraints define the boundaries and challenges that the development team needs to address.

Platform Constraints: The application is developed using Flutter, which primarily supports Android and iOS platforms. While Flutter offers experimental support for web and desktop platforms, these are not yet fully stable and may require additional development efforts. Device limitations such as screen size, performance, and operating system versions (e.g., Android 8.0+ or iOS 12+) may restrict some advanced UI/UX functionalities.

Backend Constraints: The backend is developed using Golang, known for its performance and concurrency support. However, Golang has fewer built-in libraries for high-level application features like content moderation and multimedia processing compared to languages like Python or Java. This necessitates careful selection and integration of third-party libraries. The application relies on containerized PostgreSQL databases, which require manual configuration for efficient data backup, scaling, and high availability, especially when not using managed services like AWS RDS.

Network & Deployment Constraints: The application's performance and responsiveness depend on network latency, particularly for real-time interactions such as pull requests and commenting. Deploying on AWS in specific regions may improve speed but introduces dependency on consistent internet access. Server load balancing, auto-scaling, and fault tolerance must be configured manually using AWS EC2 and related infrastructure components. Implementing Kubernetes can aid in automating deployment, scaling, and management of containerized applications, enhancing scalability and resilience.

User Interaction Constraints: Users can only read full stories after accepting terms or receiving access from the author, which may limit discoverability unless an effective onboarding and request-access system is implemented. User contributions (pull requests) must be moderated by the author, potentially slowing down collaboration if authors are inactive or unresponsive.

Security & Privacy Constraints: Protecting user-generated content from unauthorized access or tampering is crucial. Implementing secure authentication mechanisms like JWT, encrypted communication via HTTPS, and strict role-based permissions is essential. Privacy policies must comply with applicable laws, such as the General Data Protection Regulation (GDPR), to ensure proper handling of user data.

Scalability Constraints: Without managed database and messaging services, scalability depends on manual or semi-automated infrastructure management, including Docker containers, EC2 scaling, and manual database replication. As community engagement grows, moderation, storage, and notification systems may need to scale beyond initial assumptions. Utilizing Kubernetes can facilitate horizontal scaling and efficient resource management for PostgreSQL clusters, enhancing the application's ability to handle increased load.

Time & Resource Constraints: Development resources are limited, particularly if handled by a small team. Efficient time management, code reusability, and minimal viable product (MVP) planning are essential. Reliance on third-party services like AWS and external libraries introduces dependencies that may impact timelines or budgets if issues arise.

Addressing these constraints through careful planning and the adoption of appropriate technologies, such as Kubernetes for orchestration and scalability, is vital for the successful development and deployment of the collaborative storytelling application.

5.5 Use Case Model/Flow Chart/DFDS

Phase 1: Author Creates Draft: In this initial phase, the author begins by creating a new story on the StoryGram platform, entering essential details such as the title, genre, and description. The author then saves the draft and proceeds to add chapters, providing a title and content for each, which are saved as part of the draft. This stage remains private, ensuring that only the author can edit the content until it is ready for publication. The phase concludes with a confirmation that the draft has been successfully saved, setting the foundation for the subsequent publishing and collaboration stages.

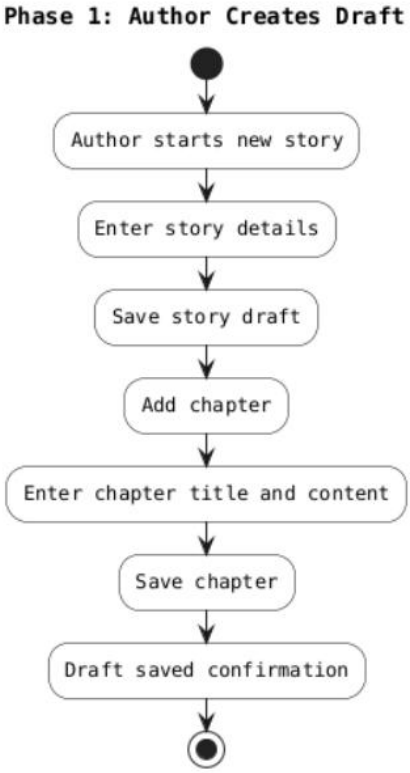

Figure 1 Author Creates Draft

Phase 2: Author Publishes Story: In this phase, the author transitions from drafting to publishing the story on the StoryGram platform. The author reviews the draft and initiates the publishing process, confirming that all chapters are complete. Upon confirmation, the system updates the chapters' status in the database and sets the story status to "published." The platform then displays a confirmation message, indicating that the story is officially published and ready for collaboration, marking the shift from a private draft to a publicly accessible narrative.

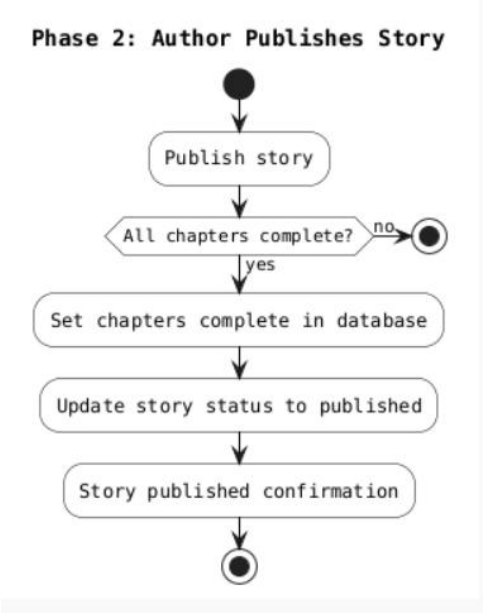

Figure 2 Author Publishes Story

Phase 3: Collaboration Request & Access: This phase begins with a collaborator viewing the published story and submitting a collaboration request to contribute. The author receives and reviews the request, deciding whether to approve or deny it. If approved, the collaborator is added to the story and notified of their access, enabling them to contribute edits and ideas. If denied, the collaborator receives a notification, and the process ends, ensuring the author retains control over who can participate in the story's development.

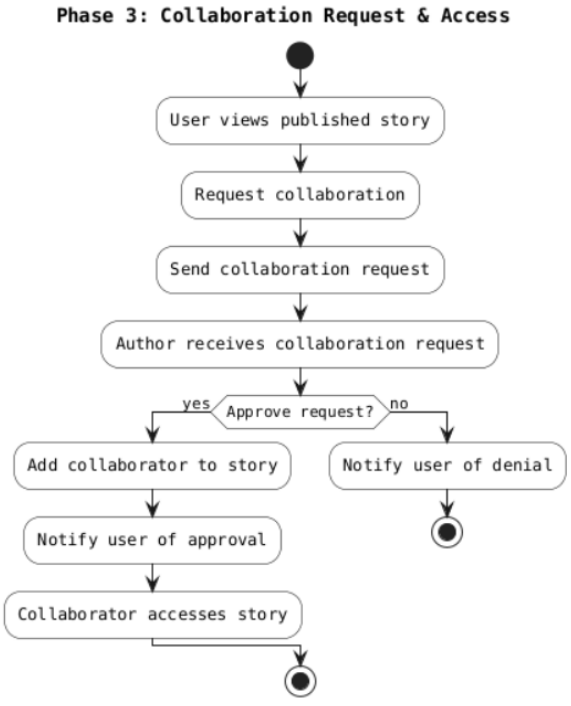

Figure 3 Collaboration Request & Access

Phase 4: Collaborator Edits & Submits: In this phase, the approved collaborator accesses the story and selects a chapter to edit on the StoryGram platform. The collaborator makes necessary changes, marks the chapter as complete or incomplete, and updates the chapter in the database. The system records the last edited timestamp, and the collaborator submits the changes, receiving a confirmation of submission. The platform notifies the author of the submitted edits, paving the way for the review process while ensuring a structured contribution workflow.

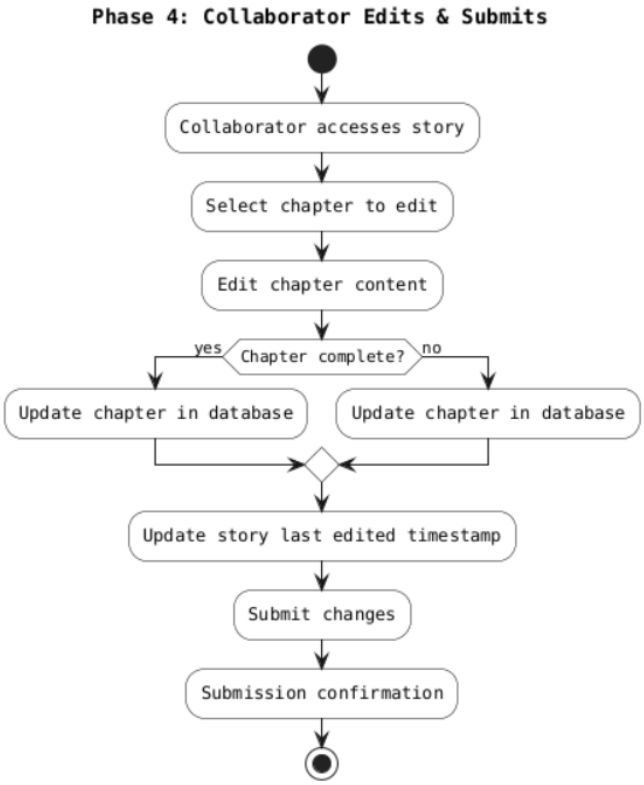

Figure 4 Collaborator Edits & Submits

Phase 5: Author Review & Merge: This phase involves the author being notified of the collaborator's submitted changes and reviewing them for approval. The author decides whether to approve or reject the edits; if approved, the changes are applied to the chapter, the last edited timestamp is updated, and the updates are finalized, making the contributions visible to all collaborators. If rejected, the author may request further edits or leave the chapter unchanged, finalizing the updates without merging, ensuring the story aligns with the author's vision.

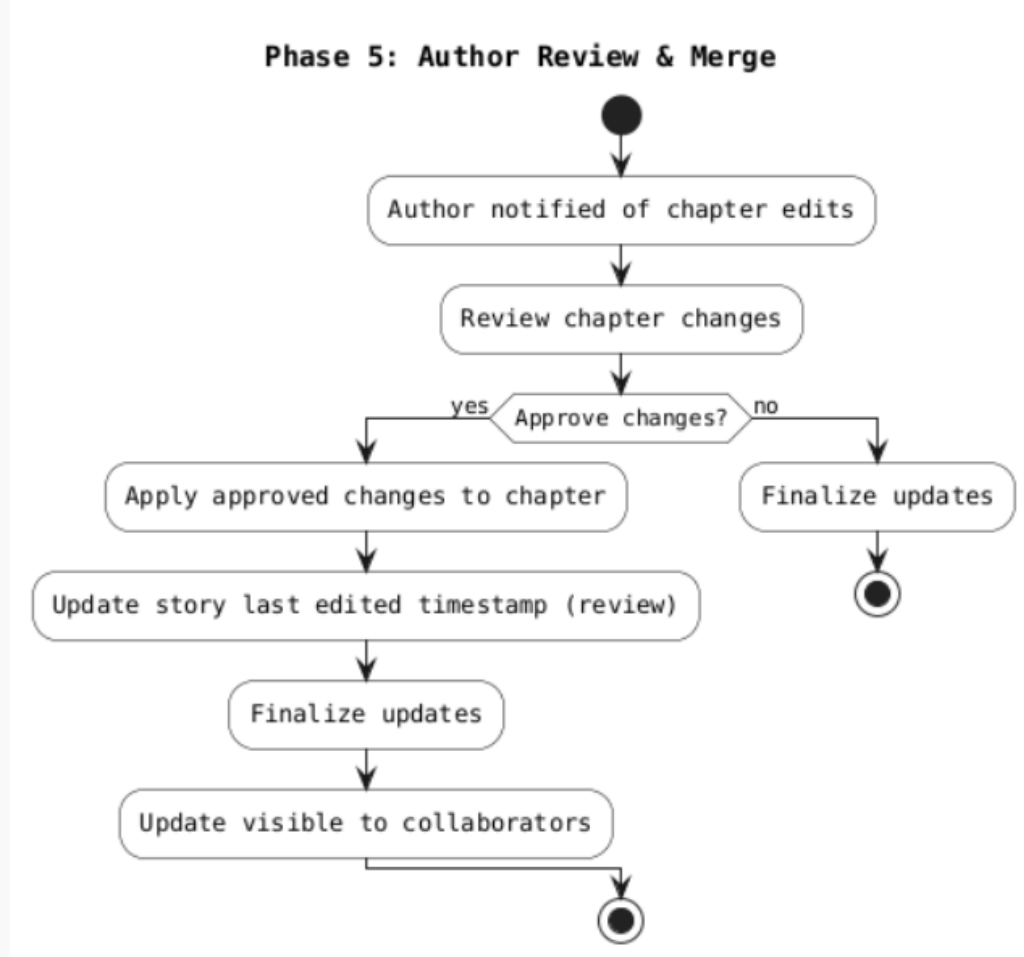

Figure 5 Author Reviews & Merge

5.6 ER Diagrams

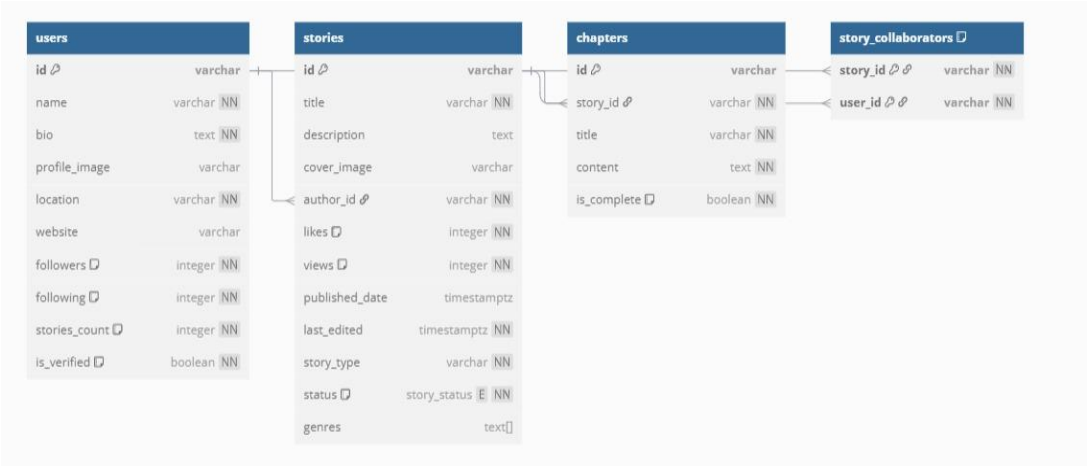

Figure 6 ER Diagram of Database Schema

Chapter 6

Result

Profile Section the profile section allows users to view their created content, showcasing individual and collaborative storytelling efforts. It also provides a space to track saved stories for later reading. Engagement metrics such as followers, following, and interaction counts are prominently displayed, offering insights into a user's activity within the platform.

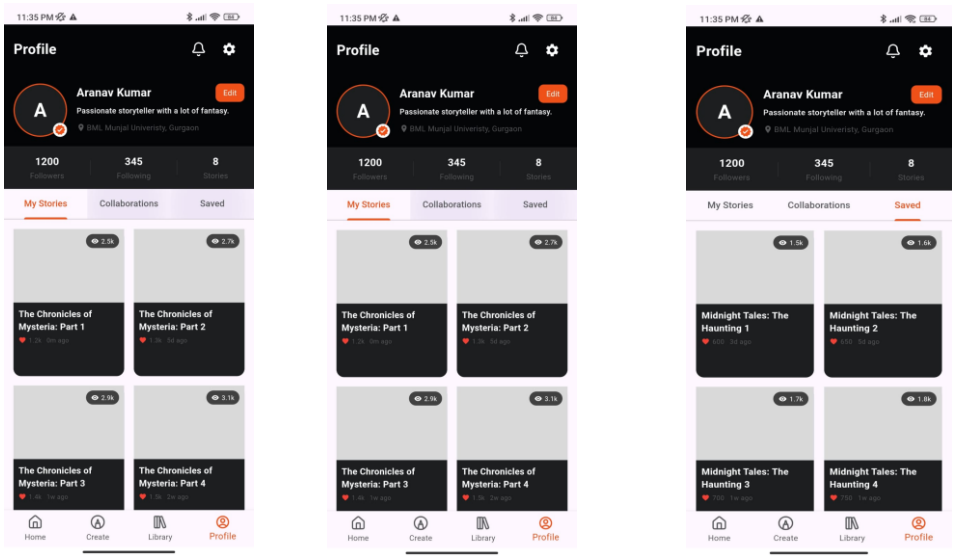

Figure 7 Profile Screen

The "Home" screen includes a search bar for exploration and a category navigation (e.g., Trending, Genres). It displays featured stories with titles, author details, brief summaries, like and view counts. A personalized recommendation section is also present. The bottom navigation bar allows access to different sections of the app.

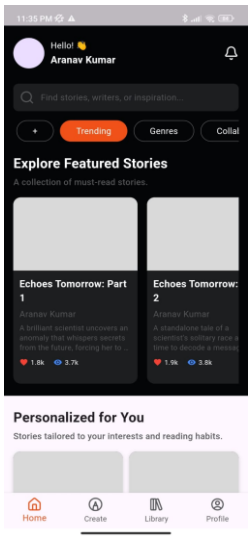

Figure 8 Home Screen

The "**Create New Story**" screen allows users to initiate new stories on the platform. It provides options for story type (single, chapter-based, or collaborative), and includes fields for essential details like title, cover image, synopsis, and genre selection. A writing area with basic formatting and a word count is provided, culminating in a "**Publish Story**" button.

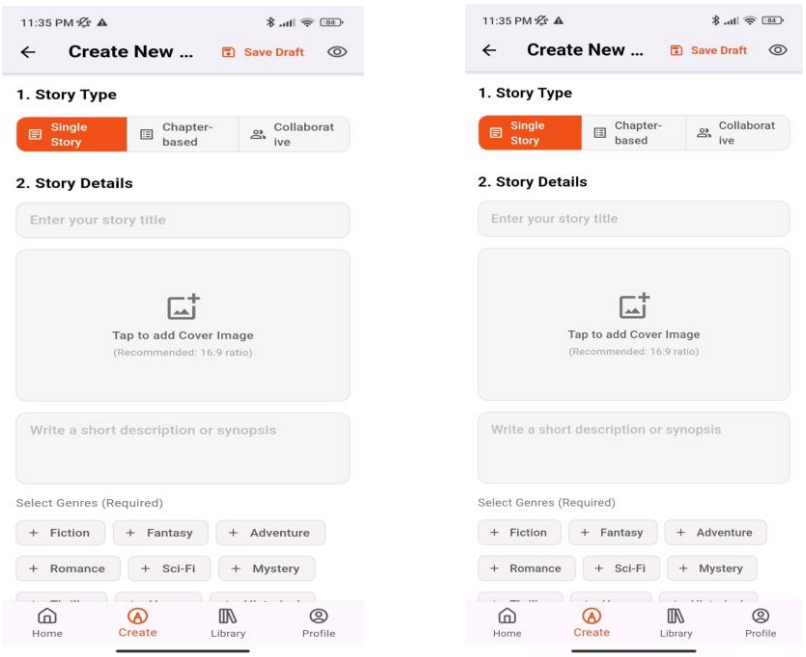

Figure 9 Create New Story Screen

The "**My Library**" section serves as a personal workspace for users to manage their stories. It provides an overview of ongoing drafts, published works, and archived content. For each story, users can see its title, current status (e.g., Draft, Published, Archived), story type (e.g., Chapter-based, Single Story), progress for chapter-based stories, and the last edited time. This section allows for easy access and organization of a user's created narratives.

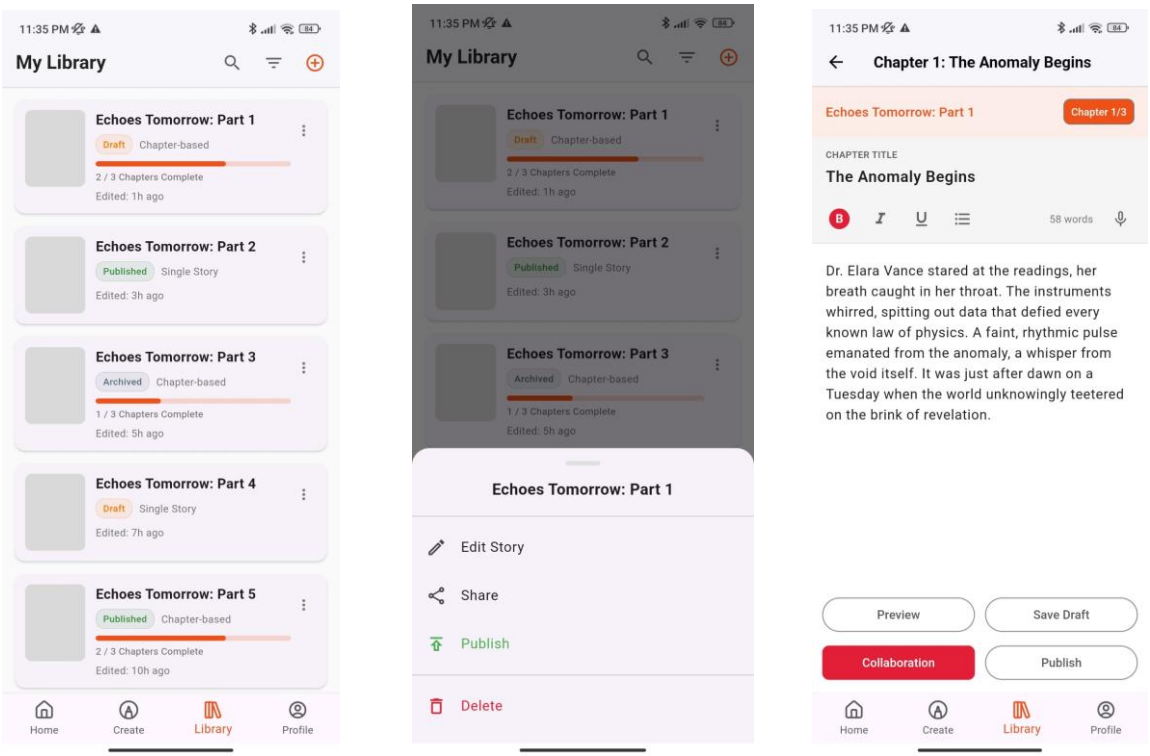

Figure 10 Library Screen

The "**Collaborations & Versions**" screen focuses on managing collaborative storytelling. It displays current collaborators for a story, indicating their role (e.g., Owner) and the date they joined. It also provides an option to invite new collaborators via email or username.

This section includes **"Collaboration Settings,"** allowing the story owner to configure how collaboration works. Options like enabling a shareable link for access requests and setting up a review system to approve edits before publishing are present. These settings give the original author control over the collaborative process.

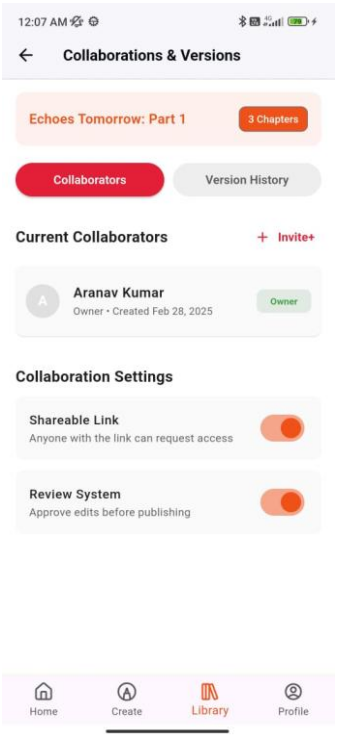

Figure 11 Collaborations & Versions Screen

The "**Version History**" tab, though currently indicating it's a "coming soon" feature, suggests the platform will allow users to track changes, compare versions, and revert to previous states of their story. This feature aims to provide transparency and control over the evolution of collaborative narratives.

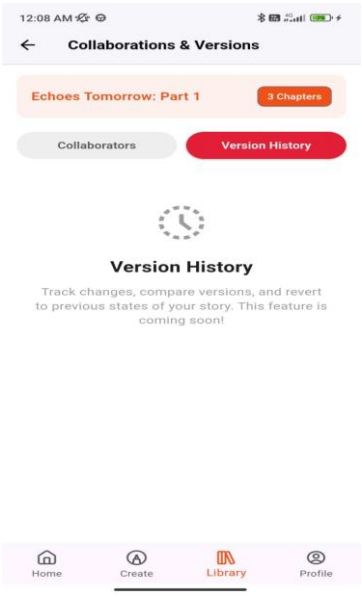

Figure 12 Version History Screen

The **"Invite Collaborator"** pop-up appears when the "Invite+" option is selected. It prompts the user to enter the email or username of the person they wish to invite to collaborate on the story, streamlining the process of adding contributors.

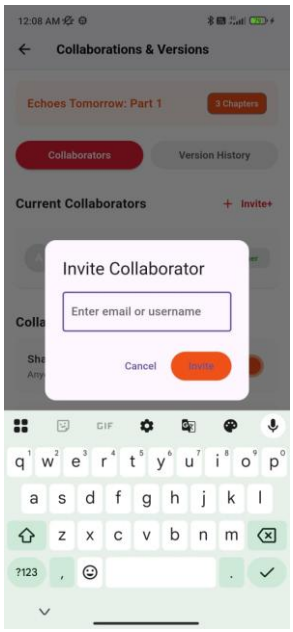

Figure 13 Invite Collaborator

Chapter 7

7.1 Conclusion

The Collaborative Story Writing Platform marks a significant step toward reimagining how creative writing can be experienced and enhanced through technology. By drawing inspiration from code collaboration platforms like GitHub and combining it with the emotional and expressive world of storytelling, this application creates a new digital space where creativity thrives through community. The platform successfully bridges the gap between authors and readers by introducing structured collaboration via pull requests, gated content access, blurred previews, and author-controlled contributions. This not only protects the integrity of the story but also encourages meaningful reader engagement. With features such as story voting, search discoverability, reviews, and community feedback mechanisms, the platform goes beyond traditional reading apps by fostering an active, participatory ecosystem. The use of Flutter ensures a smooth and responsive cross-platform experience, while backend technologies like Golang and deployment on AWS make the system scalable and robust. Overall, the project delivers a novel, interactive storytelling experience that empowers writers and engages readers like never before.

7.2 Future Scope

While the current implementation lays the foundation for a dynamic collaborative writing platform, there is vast potential for future enhancements. One key area is the introduction of AI-assisted writing suggestions, which can help both authors and collaborators maintain tone, improve grammar, and enhance narrative flow. Additionally, a version control system specifically tailored for story drafts can offer deeper insight into how stories evolve over time, allowing authors to track changes and revert if needed. Gamification elements, such as badges for top contributors or popular stories, can further motivate user engagement. To support a global user base, implementing multi-language support and localization features would significantly expand the platform's reach. Enhanced analytics dashboards for authors—tracking reads, contributions, and reader engagement—can provide actionable feedback. Finally, integrating blockchain-based copyright protection could offer authors stronger ownership control and transparent attribution. As the platform evolves, these future developments could transform it from a collaborative tool into a full-fledged ecosystem for interactive storytelling.

Bibliography

Certainly! Here is the bibliography with direct links to the referenced papers:

1. Montanelli, S., & Ruskov, M. (2023). *A Systematic Literature Review of Online Collaborative Story Writing*. Human-Computer Interaction – INTERACT 2023. Springer. https://www.researchgate.net/publication/373380513_A_Systematic_Literature_Review_of_Online_Collaborative_Story_Writing
2. Wilson, D. (2023). *A Collaborative Story Writing Project Using Google Docs and Face-to-Face Collaboration*. Canadian Journal of Learning and Technology, 49(3), 1–21. https://cjlt.ca/index.php/cjlt/article/view/28174
3. Robayo Luna, A. M., & Hernández Ortiz, L. S. (2013). *Collaborative Writing to Enhance Academic Writing Development Through Project Work*. HOW, 20, 130–148. https://files.eric.ed.gov/fulltext/EJ1127904.pdf
4. Li, Y. (2023). *The Effect of Online Collaborative Writing Instruction on Enhancing Writing Performance, Writing Motivation, and Writing Self-Efficacy of Chinese EFL Learners*. Frontiers in Psychology, 14, 1165221. https://www.frontiersin.org/articles/10.3389/fpsyg.2023.1165221/full(CoLab).

Plagiarism Report

PRJ 3 Report.pdf

ORIGINALITY REPORT

6%	4%	1%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to BML Munjal University Student Paper	4%
2	ir.juit.ac.in:8080 Internet Source	<1%
3	pmc.ncbi.nlm.nih.gov Internet Source	<1%
4	"Human-Computer Interaction – INTERACT 2023", Springer Science and Business Media LLC, 2023 Publication	<1%
5	redwerk.com Internet Source	<1%
6	www.coles-miller.co.uk Internet Source	<1%
7	Submitted to Canterbury Christ Church University Student Paper	<1%
8	Submitted to University of Newcastle Student Paper	<1%
9	www.docdroid.net Internet Source	<1%
10	www.viavisolutions.com Internet Source	<1%