American International University-Bangladesh (AIUB)

Department of Computer Science

Faculty of Science &Technology (FST)

Summer 24-25

Crowd Source bug tracking and feature suggestion platform

Software Requirement Engineering

Sec: **A**

Project submitted

By

FARDIN-AL-SEZAN (22-46868-1)
MD. NURE AMIN (21-45636-3)
ANINDA SAHA SHOUDHA (21-44866-2)
SHANAWAZ CHOWDHURY (20-44350-3)

Contents

1. PROBLEM DOMAIN

1.1Background to the Problem

In today's fast-paced software industry, ensuring reliability and usability is a constant challenge. Although internal testing is done, many bugs and usability issues only appear when real users interact with the product in diverse environments. At the same time, end-users often have valuable suggestions for new features but lack a proper channel to share them with developers.

The root cause of this challenge is the absence of a simple, centralized, and transparent platform for collecting and prioritizing user feedback. Existing tools like Jira or Bugzilla are powerful but too complex for non technical users, which results in valuable insights being lost in scattered channels such as emails or social media.

This issue is critical because unresolved bugs and ignored feedback reduce user satisfaction, damage product reputation, and weaken competitiveness. To meet user expectations for timely fixes and improvements, a community driven bug tracking and feature suggestion system is essential for improving product quality and trust.

Software reliability challenges: While internal testing is performed, many bugs and usability issues only surface when real users interact with the product in diverse environments.

Lack of a feedback channel: End users often have valuable suggestions for new features but there is no simple, centralized platform for them to share feedback directly with developers.

Complexity of existing tools: Existing platforms like Jira or Bugzilla are powerful but too complex for non technical users. As a result, important insights are lost in scattered channels such as emails or social media.

Critical consequences: Unresolved bugs and ignored feedback lead to reduced user satisfaction, damage to product reputation, and loss of competitiveness in the market.

Need for transparency and community involvement: A user friendly, transparent platform is essential to engage the community, prioritize issues effectively and build trust between users and developers.

1.2 Solution to the Problem

We propose a Crowd-Source Bug Tracking and Feature Suggestion Platform that enables users to report bugs, suggest features, and vote on issues. This approach directly involves the community in development, ensures transparency, and helps teams focus on the most critical tasks. The solution is feasible with modern web technologies that support scalability, security, and usability, aligning well with business objectives.

The platform will act as a centralized hub for managing feedback. Its purpose is to enhance software quality and user satisfaction by providing a structured process for reporting and prioritizing issues. Key benefits include faster bug detection, efficient resource allocation, and stronger trust between users and developers. The main goals are to simplify reporting, prioritize effectively, and reduce resolution time.

Existing tools like Bugzilla, Jira, and GitHub Issues already address parts of this problem, but they are often too complex or lack community driven prioritization. Our solution bridges this gap by combining simplicity with user participation making it more accessible while still supporting structured decision making.

Faster Bug Detection: With a larger pool of users actively reporting issues, bugs can be identified and reported more quickly than through traditional methods.

Efficient Resource Allocation: By allowing the community to vote on issues, development teams can focus on what matters most to the users ensuring that resources are allocated to the most critical bugs and most desired features.

Reduced Resolution Time: A clear, prioritized list of issues helps streamline the development workflow and reduce the time it takes to fix bugs and implement new features.

Increased Transparency & Trust: Involving the community in the process fosters greater transparency and builds stronger trust between users and developers. Users feel heard and are more invested in the product's success.

2. SOLUTION DESCRIPTION

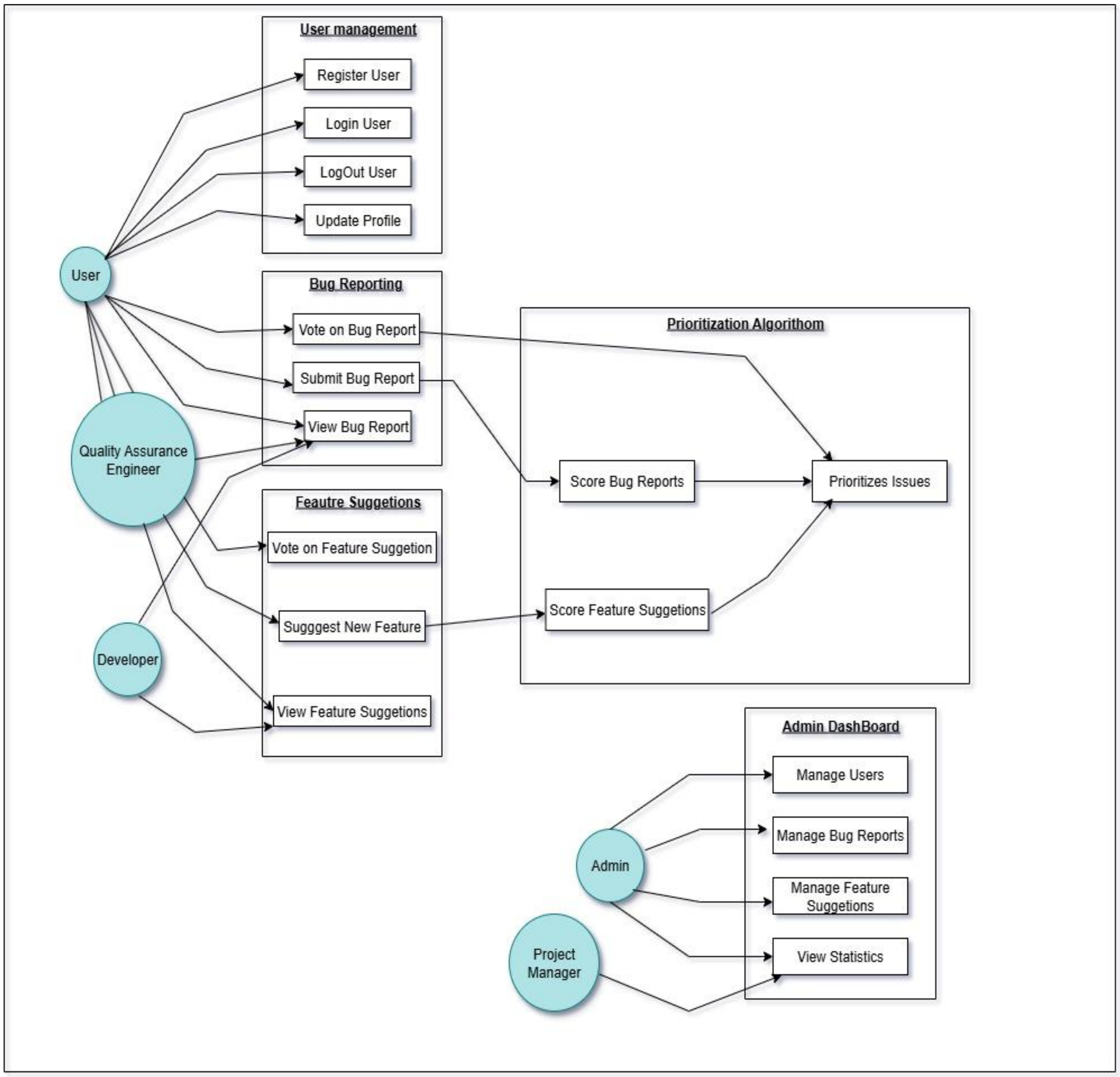
2.1System Features

Feature Name	User Roles	Functional Requirement	Quality Attributes
User Registration & Authentication	User, Admin	FR-1.1: Users can register with email/password. FR-1.2: Users can log in and log out securely. FR-1.3: Password reset via email.	Security: Password hashing, JWT authentication. Usability: Simple login/registration UI. Reliability: 99.9% uptime.
Bug Report Submission	User	FR-2.1: Users can submit bug reports with severity, urgency, environment, and attachments. FR-2.2: Duplicate detection when submitting reports	Usability: Predefined templates for reporting. Data Integrity: Validations on input & attachments. Performance: Response <2s.
Feature Suggestion	User	FR-3.1: Users can submit feature requests. FR-3.2: Suggestions can include description, priority, and potential benefits.	Transparency: Suggestions are publicly visible. Scalability: Able to handle growing number of suggestions.
Voting System	User	FR-4.1: Users can up/downvote bug reports. FR-4.2: Users can upvote feature appeal	Fairness: One vote per user per item. Auditability: Voting history stored securely.

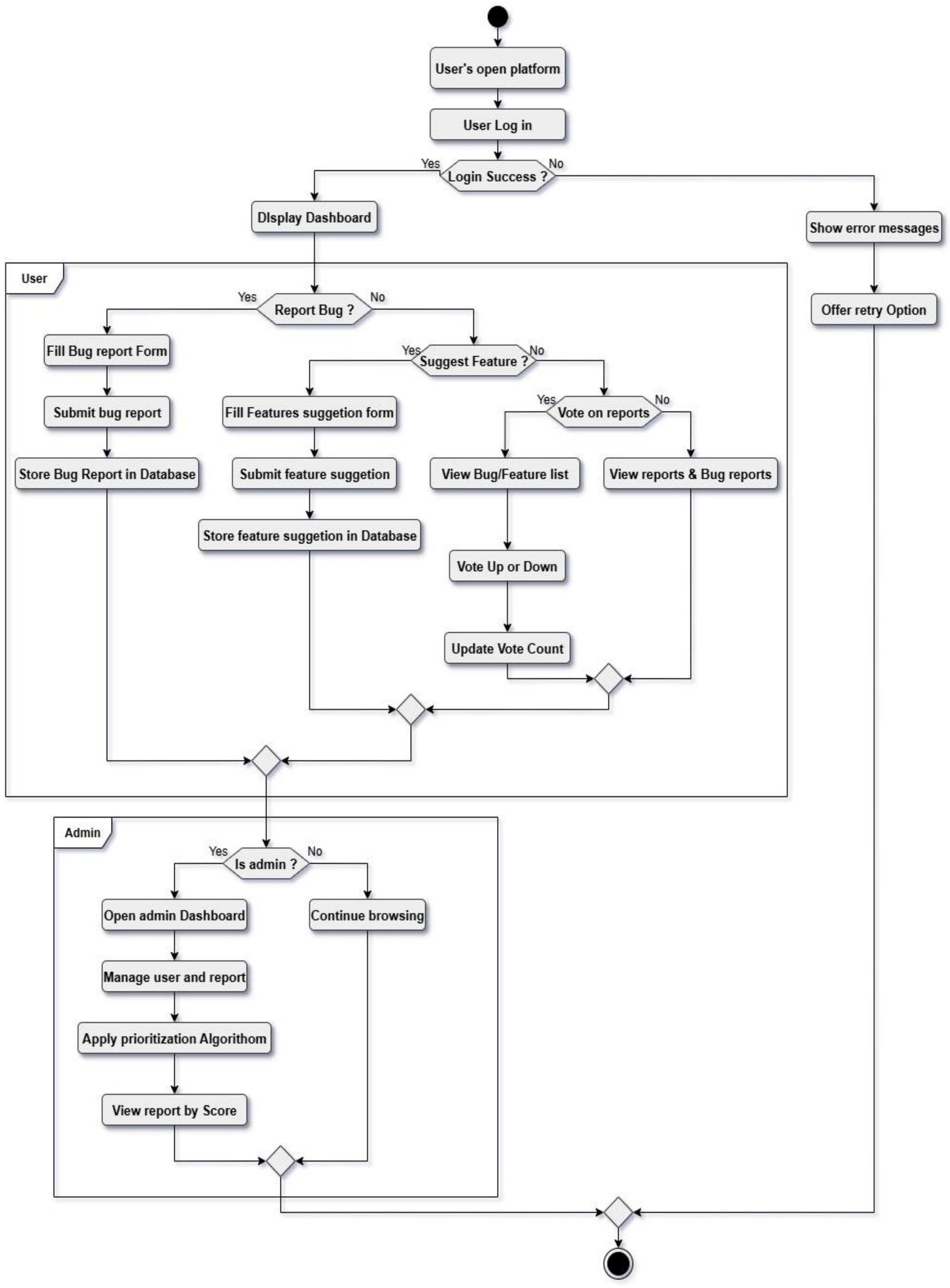
Admin Dashboard	Admin	FR-5.1: Admin can view, triage, and assign bug reports. FR-5.2: Admin can update issue status (Open, In-progress, Resolved, Closed). FR-5.3: Admin can moderate feature requests.	Security: Role-based access control (RBAC). Auditability: Every action logged with admin ID.
Search & Duplicate Spying	User, Admin	FR-6.1: Users can search existing reports/suggestions. FR-6.2: System flags potential duplicates during submission.	Performance: Search results under 2 seconds. Reliability: Accurate duplicate detection.
Commenting & Discussion	User, Admin	FR-7.1: Users can comment on bug reports & feature requests. FR-7.2: Admin can moderate inappropriate comments.	Transparency: Publicly visible discussions. Moderation: Admin can remove abusive content.

2.2UML DIAGRAMS

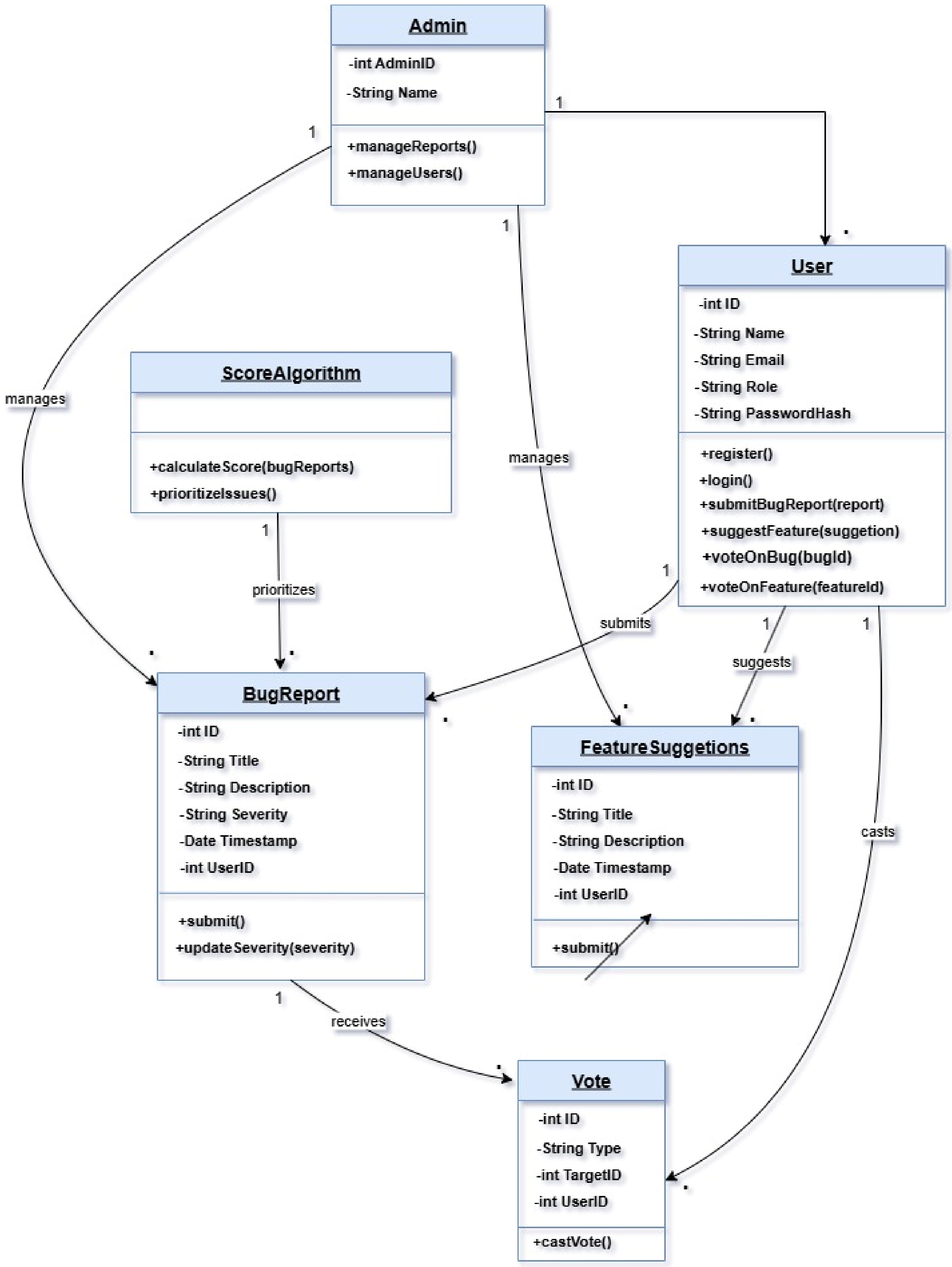
Use-case diagram:



Activity diagram:



Class diagram:



3. Social Impact

The platform will benefit society by giving users an active role in improving the software they use daily. It ensures transparency and trust by allowing people to see how their feedback is addressed. Developers will be able to respond faster to real issues, improving product quality and user satisfaction. This collaboration between users and developers encourages responsible software practices. In the long run, it helps businesses, startups, and communities build more reliable and user friendly digital solutions that support social and economic growth.

3.1 Empowering End Users

- Provides users with an active role in shaping the software they use.
- Ensures their feedback is heard, reducing frustration and increasing satisfaction.
- Motivates users by letting them see their contributions directly improve the product.
- Encourages loyalty as users feel valued and involved in decision making.

3.2 Encouraging Community Collaboration and Transparency

- Builds a bridge between developers and users by creating an open channel for discussion.
- Increases accountability as users can see how their feedback is processed and prioritized.
- Encourages peer to peer interaction where users can upvote or validate each other’s feedback.
- Creates a transparent roadmap visible to all stakeholders.
- Strengthens trust by showing clear progress and developer responsiveness.

3.3 Improving Accessibility Across Diverse Environments

- Real users from different devices, operating systems, and networks can report issues.
- Helps developers detect bugs that internal testing teams might miss.
- Ensures inclusivity by addressing accessibility issues faced by differently abled users.
- Expands test coverage beyond limited QA environments.
- Improves adaptability of the software across regions and cultures.

3.4 Enhancing Digital Trust and Product Quality

- Transparent bug reporting and resolution strengthens user confidence in software.
- Results in higher-quality, more reliable applications that meet real needs.

- Provides data-driven insights into recurring issues and critical bugs.
- Reduces security risks by quickly identifying and addressing vulnerabilities.
- Builds a reputation for accountability and continuous improvement.

3.5 Supporting Startups and Businesses

- Lowers the barrier for startups and small teams by providing a ready-to-use feedback hub.
- Promote responsible software practices that improve competitiveness in the industry.
- Saves costs on large scale testing teams by leveraging community efforts.
- Accelerates product improvement cycles through rapid feedback collection.
- Provides startups with valuable user driven market insights for growth.

4. Development Plan with Project Schedule

The prosperous creation and deployment of the Crowd Source bug tracking and feature suggestion platform takes a setup, disciplined and agile development plan. This particular plan outlines the methodology, timeline, phases and team structure are essential to changing the idea into a scalable and functional product.

4.1 SDLC Phases:

1. **Requirement Analysis** – Collect needs from users and stakeholders.
2. **Design** – Prepare UML, ER diagrams, and UI mockups.
3. **Implementation** – Develop backend (Node.js/Express) and frontend (Next.js).
4. **Testing** – Unit, integration, load, and security testing.
5. **Deployment** – Host on cloud with monitoring.
6. **Maintenance** – Continuous updates, bug fixes, and optimizations.

4.2 Development Methodology: Agile-Scrum Approach

The platform will be developed using the Agile-Scrum process to ensure flexibility, rapid iteration, and alignment with project goals. This approach is ideal because feature priorities can be influenced by user feedback and evolving market needs.

- **Sprint Duration:** Fixed two week sprints.
- **Sprint Ceremonies:** Daily stand ups, sprint planning, sprint review and retrospective.
- **Sprint Planning:** Selection of the highest priority features from the backlog.
- **Daily Stand-ups:** Short team meetings to track progress and obstacles.

- **Sprint Review:** Demo of completed features to stakeholders for feedback.
- **Sprint Retrospective:** Reflection meeting to improve processes for the next sprint.
- **Product Backlog**: Centralized list of all system features, user stories and tasks prioritized by business value and user needs.

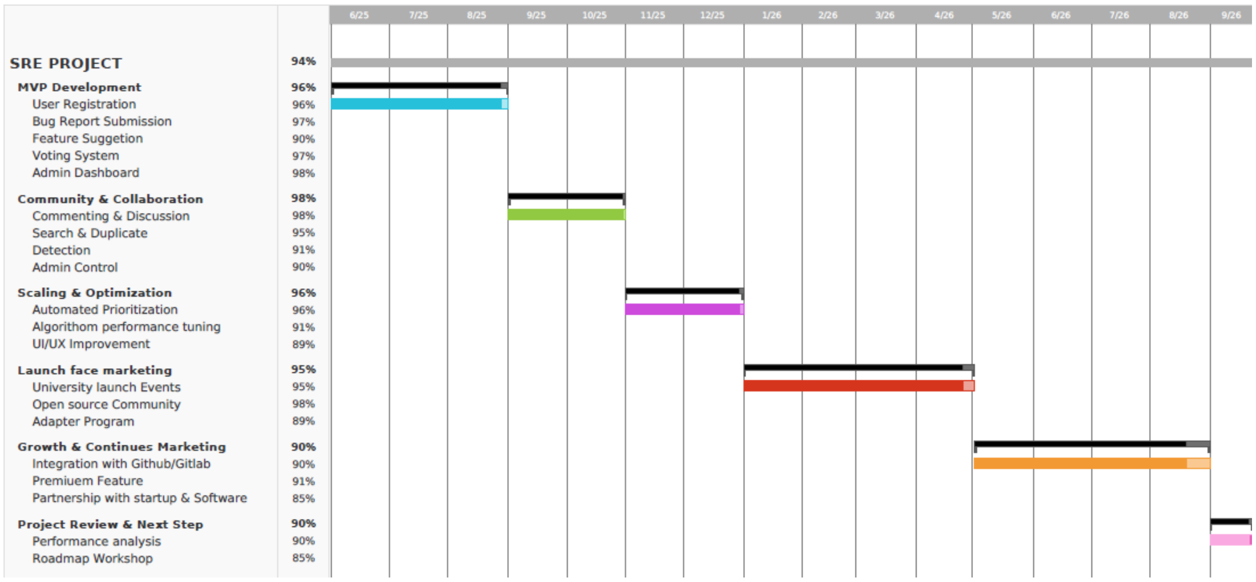
4.3 Phased Development Roadmap

The project will be built in multiple phases, starting with the Minimum Lovable Product (MLP) and expanding over time. A box has given below :

Phase	Tasks	Starte Date	End date	Duration
Phase 1: MVP Development	User Registration & Authentication Bug Report Submission Feature Suggestion Voting System Basic Admin Dashboard	1 June, 2025	31 August, 2025	3 Months
Phase 2: Community & Collaboration	Commenting & Discussions Search & Duplicate Detection Enhanced Admin Controls	1 September, 2025	30 October, 2025	2 Months
Phase 3: Scaling & Optimization	Automated Prioritization Algorithm Performance Tuning UI/UX Improvements	1 November, 2025	31 December,2025	2 Months
Phase 4: Launch-Phase Marketing	University Launch Events Open-Source Community Onboarding Early Adopter Program	1 January, 2026	30 April, 2026	4 Months

Phase 5: Growth & Continuous Marketing	Integration with GitHub/GitLab Premium Features (Analytics, AI Duplicate Detection) Partnerships with Startups & Software Houses	2 May, 2026	31 August, 2026	4 Months
Phase 6: Project Review & Next Steps	Performance Analysis Roadmap Workshop for v2.0	1 September, 2026	22 September, 2026	3 Weeks

4.4 Sample Project Schedule (Gantt Chart):



5. Marketing Plan

5.1 Short-Term Plan:

1. **University and Open Source Launch:** Introduce the platform in universities and open source developer groups to attract early contributors and testers.
2. **Free Tier for Small Teams:** Offer free access to startups and student projects to encourage adoption.
3. **Beta Testing Program**: Invite a select group of developers and users to test the platform and provide feedback before the full launch.
4. **Social Media Awareness:** Create awareness through LinkedIn, Twitter, and tech-related Facebook groups by highlighting benefits like transparency and collaboration.

5.2 Long-Term Plan:

1. **Partnerships with Startups and Software Houses**: Collaborate with local software companies to integrate the platform into their development process.
2. **Integration with Developer Tools**: Provide GitHub/GitLab integration so issues can sync directly with CI/CD pipelines.
3. **Premium Subscription Features**: Offer analytics dashboards, team management and AI based duplicate bug detection as paid upgrades.
4. **Conference and Workshop Presence:** Present at technology conferences and workshops to showcase the platform to industry professionals.

5.3 Continuous Plan:

1. **Active Community Building:** Run forums, Q&A sections and discussion groups where developers and users can collaborate.
2. **Content Marketing:** Publish blog posts, tutorials and case studies demonstrating how user feedback improved real products.
3. **Social Media Engagement:** Share platform updates, success stories and highlight top contributors using hashtags like #BugSquashBD.
4. **Loyalty & Recognition Programs:** Reward active users and contributors with badges, recognition, or free premium months to keep them engaged.
5. **Regular Feedback Analysis:** Collect, categorize and analyze feedback trends to identify recurring issues and prioritize improvements.
6. **Integration with Popular Tools:** Allow users to connect the platform with GitHub, Jira, or Slack for seamless workflow integration.

6. Cost and Profit Analysis

A thorough cost and profit analysis is essential to determine the financial viability of the Crowd Source Bug Tracking and Feature Suggestion Platform and to secure investor confidence. This analysis includes development and marketing costs, followed by a projection of profitability.

6.1 Project Estimation

To estimate the project's requirements, we apply the Constructive Cost Model (COCOMO) an algorithmic software cost estimation method. Based on the project's scope a community driven web platform with user management, voting and bug tracking we classify it as an organic software project.

- Source Lines of Code (SLOC): 20,000 (estimated for core features)
- Effort Coefficient: 2.5
- Project Complexity (P): 1.25
- Time Coefficient (T): 0.40

Effort (Person-Months)

$$\text{PM} = 2.5 \times \left(\frac{20000}{1000}\right)^{1\cdot 25}$$

$$\text{PM} = \ 2 \cdot 5x(20)^{1\cdot 25} \approx 2.5\times 42.3 \approx 105.8 \text{ person-months}$$

Development Time (Months)

$$\text{DM} = 2 \cdot 5 \times (PM)^{0\cdot 40}$$

$$\text{DM} = 2 \cdot 5 \times (105 \cdot 8)^{0\cdot 40} \approx 14 \cdot 3 \approx 14 \text{ months}$$

Personnel Requirement

$$\text{ST} = \frac{\text{PM}}{\text{DM}} = \frac{1058}{143} \approx 7 \cdot 4 \approx 7$$

6.2 Cost Analysis

SL	Cost Item	Total Hours	Hourly Rate (BDT)	Resource Count	Total Cost (BDT)
1	Develpers (4)	2464	800	4	7,88,480
2	UI/UX Design (shared)	2464	700	1	1,72,480
3	QA & Testing (shared)	2464	700	1	1,72,480
4	Project Management (rotational)	2464	900	1	2,21,760
5	Cloud Hosting & Tools (14 months × 20,000 BDT/months)	-	-	-	2,80,000
6	Security Audit & Miscellaneous	-	-	-	1,00,000
7	Marketing & Community Outreach (Pre/Post Launch)	-	-	-	3,00,000

Hours ≈ 14 months × 22 working days/month × 8 hours/day.

Total Estimated Development Cost = 18,34,000 BDT (≈ 18.3 Lakh BDT)

6.3 Profit Analysis

The profit analysis for Crowd source Bug tracking is based on its primary revenue stream a commission fee on every sale made through the platform.

Assumptions:

- Subscription Price: 1,000 BDT/user/month
- Target Paying Users (Year 1): 400
- Monthly Revenue: 400 × 1,000 = 4,00,000 BDT
- Yearly Revenue: 4,00,000 × 12 = 48,00,000 BDT

First-Year Profit Projection:

$$\text{Profit} = \text{Total Revenue} - \text{Total Cost}$$
$$\text{Profit} = 48{,}00{,}000 - 18{,}34{,}000 = 29{,}66{,}000\text{BDT}$$

7. Reference

[1] https://www.researchgate.net/publication/262528436_Cloud Based_Software_Crowdsourcing

[2] https://arxiv.org/pdf/1810.00125

[3] https://onlinelibrary.wiley.com/doi/10.1002/smr.2705

[4] Wikinomics: How Mass Collaboration Changes Everything – Don Tapscott, Anthony D. Williams (Expanded Edition, 2008)

[5] The Wisdom of Crowds – James Surowiecki (1st Edition, 2004)