SB Works

Freelancing Platform

SDLC Phase 7

MAINTENANCE & DOCUMENTATION REPORT

Project Name	SB Works — Online Freelancing Platform
Phase	Phase 7: Maintenance and Documentation
Prepared By	I. Sai Ganesh — Lead Developer
Date	February 2024 — Ongoing
Status	Active Maintenance

1. Introduction to Maintenance

1.1 What is the Maintenance Phase?

Maintenance is the final and longest-running phase of the SDLC. After a project is deployed and in use, it never truly "stops" — it continues to evolve. Users report bugs that slipped through testing, the technology ecosystem changes (libraries release new versions, security vulnerabilities are discovered), and new requirements emerge as users interact with the system.

For SB Works, the maintenance phase began immediately after deployment. Even during the deployment process itself, several issues came up (MongoDB Atlas connection errors, Windows environment issues) that required fixes. These are classic "post-deployment maintenance" activities. This report documents what was maintained, what documentation was produced, and what future improvements are planned.

> ■ Maintenance Philosophy: Good maintenance is not just about fixing bugs. It includes keeping documentation up to date, upgrading dependencies regularly, monitoring for security issues, and planning new features based on user feedback. A well-maintained project can serve its users for years without a complete rewrite.

1.2 Types of Maintenance Performed

Maintenance Type	Description	Example in SB Works
Corrective	Fixing bugs found after deployment	MongoDB connection string format issues, Windows path errors
Adaptive	Adapting to changes in environment	Switching from local MongoDB to Atlas, updating .env format
Perfective	Improving performance or usability	Adding dev mode OTP, flexible phone validation
Preventive	Preventing future problems	Writing comprehensive troubleshooting guide, documenting all .env variables

2. Post-Deployment Bug Fixes

2.1 Issues Found After Deployment

The following issues were discovered and resolved during the maintenance phase. These are separate from the bugs found in Phase 5 testing — these arose specifically from real-world deployment conditions:

Issue	When Found	Root Cause	Fix Applied
Backend crashed on missing Twilio creds	First deployment attempt	Hard-coded throw if SID missing	Added dual-mode: dev prints OTP, prod uses Twilio
Phone validation rejected valid numbers	First user registration attempt	Regex only allowed US +1 format	Changed to flexible 7-20 char validation
Status 1 users got 403 after login	Second test user login	Middleware blocked non-approved users too aggressively	Allow status 1 and 2, only block status 0
Atlas URI with port caused parse error	Cloud DB setup	User added :27017 to SRV string	Documented SRV format, added to troubleshooting guide
Username with @ caused auth failure	Atlas connection	Special chars in username break URL	Documented encoding, suggested new user creation
File downloads returned 404	First work submission test	Frontend used direct URL not through proxy	Fixed download URL to use Vite proxy path

2.2 Dependency Security Notices

During npm install, two dependency warnings were shown. Both were reviewed and their impact assessed:

Warning	Package	Assessment	Action Taken
Deprecated: scmp@2.1.0	Indirect dependency	No direct usage; low risk	Monitored — will update when parent package updates
Vulnerable: multer@1.4.5	File upload library	Multer 1.x has known CVEs fixed in 2.x	Noted for upgrade in next maintenance cycle; current use is low-risk

3. Documentation Produced

3.1 Documentation Index

During and after the project, the following documentation was created to support maintenance, onboarding of new developers, and academic reporting:

Document	Purpose	Pages	Format
README.md	Quick start guide, setup in 3 steps, login instructions	2 pages	Markdown
Phase 1: Project Planning Report	This document — planning decisions, timeline, risks	6 pages	PDF
Phase 2: Requirements Analysis Report	All functional and non-functional requirements	6 pages	PDF
Phase 3: System Design Report	Architecture, database schemas, API design, UI design	7 pages	PDF
Phase 4: Implementation Report	Coding approach, feature implementation details	7 pages	PDF
Phase 5: Testing Report	Test cases, results, bugs found and fixed	6 pages	PDF
Phase 6: Deployment Report	Setup guide, configuration reference, troubleshooting	5 pages	PDF
Phase 7: Maintenance Report	Post-deployment fixes, future plans, lessons learned	6 pages	PDF
Project Documentation PDF	Complete technical reference: all models, APIs, routes	26 pages	PDF
Implementation Guide PDF	Screen mockups for every page + API with sample data	29 pages	PDF

3.2 Code Comments and Readability

All major functions and middleware in the backend were written with clarity in mind. Variable names are descriptive and follow consistent conventions:

- Controller files end with .controller.js — instantly recognizable as business logic files.
- Model files match their collection name (user.js, project.js) — easy to find in the folder.
- All route handlers follow the same pattern: middleware chain → controller function, with no business logic in routes.
- The seed.js file contains clear comments explaining what each seeded record is for.
- The .env.example comments (in the template version) explain every variable with an example value.
- Axios interceptors in httpService.js are commented to explain the auto-refresh token logic.

4. Future Improvements and Project Reflection

4.1 Planned Future Improvements

The following features are out of scope for the current version but are planned for future development cycles:

Feature	Priority	Estimated Effort	Reason Deferred
Payment gateway integration (Stripe)	High	1 week	Requires Stripe account, business registration
Email notifications (Nodemailer)	Medium	3 days	Twilio SMS chosen as primary; email is secondary
Upgrade Multer to v2.x	Medium	1 day	Address security warnings in npm audit
Mobile responsive design	Medium	1 week	Bootstrap provides basic responsiveness; full mobile needs work
Freelancer portfolio section	Medium	3 days	Skills field exists; full portfolio upload deferred
Project milestone system	Low	2 weeks	Complex feature; basic approval system sufficient for v1
Social login (Google OAuth)	Low	3 days	OTP login is sufficient and more secure for this context
Admin analytics dashboard with charts	Low	4 days	Basic stats shown; detailed charts deferred
Full-text search with MongoDB Atlas Search	Low	2 days	Basic keyword search implemented; full-text is optional
Dispute resolution system	Low	2 weeks	Complex workflow; not needed for v1

4.2 Lessons Learned — Overall Project Reflection

Looking back across all seven phases, the following lessons were the most valuable from this project:

Lesson	Detail
Plan before coding saves 10x the time later	The database schema designed in Phase 3 required almost no changes during implementation. Good planning prevents expensive rework.
Environment issues are the biggest time sink	More time was spent on MongoDB Atlas connection strings and Windows PowerShell issues than on any single feature. Document these early.
Dev mode fallbacks are essential	The Twilio dev mode (printing OTP to console) turned what would have been a blocking issue into a minor note. Always build fallbacks for paid external services.
Test with real data early	Running the seed script and logging in as a real user in Week 2 (not Week 3) would have caught the middleware bug earlier.

Keep scope tight for v1	Resisting the temptation to add payment processing and email in v1 allowed the core features to be done well. A good v1 is better than a mediocre v1.5.
Role-based design from day 1 matters	Designing the three-role system (Admin/Owner/Freelancer) before writing the first line of code meant the entire architecture supported it cleanly.
Real-time features are easier than expected	Socket.io's documentation and API are excellent. Adding real-time chat and notifications took 2 days, not the feared 2 weeks.

■ Project Completion Summary: SB Works was successfully completed across all 7 SDLC phases. 39 API endpoints were implemented. 15 frontend pages were built. 10 bugs were found and fixed. All planned features in scope were delivered. The project is working, documented, and ready for demonstration or further development.

4.3 Final Project Statistics

Metric	Value
Total SDLC Phases Completed	7 of 7
Total API Endpoints	39 endpoints across 9 route groups
Total React Pages	15 pages across 3 role dashboards
Database Collections (Models)	8 MongoDB collections
Real-time Socket.io Events	6 events (join, send, receive, notifications)
Bugs Found During Testing	10 bugs
Bugs Fixed	10 of 10 (100% resolution rate)
Post-deployment Issues Fixed	6 additional issues
Total Documentation Pages	70+ pages across all reports
Total Development Time	Approximately 35 days across all phases
Lines of Code (Approximate)	3,500+ lines (backend + frontend combined)
Technologies Used	12 (MongoDB, Express, React, Node, Socket.io, Bootstrap, MUI, JWT, Multer, Joi, Vite, Axios)