Software Development Team
Mawsters
Nanyang Technological University
50 Nanyang Ave, Singapore 639798

SHELVD
Release Plan
Version 1.3

Document Number: BC-RP-001
Contract Number: BC-24-C-0098

APPROVALS

Approving Authority:

Printed Name/ Position Title	**Signature**	**Date**	**Phone Number**
Nyan Maw Htun/ Project Manager	Nyan	27-03-04	98788517

Release Manager Approval:

Printed Name/ Position Title	**Signature**	**Date**	**Phone Number**
Cheong Yu Qing, Crystal/ Release Manager	CC	27-03-04	98912889

REVISION HISTORY

Version	Date	Organization/Point of Contact	Description of Changes
1.0	25/03/2024	Mawsters/ Nyan Maw Htun	Baseline Version, formatting, initial content
1.1	26/03/2024	Mawsters/ Arun Ezekiel	Details in sections 1-4, formatting
1.2	27/03/2024	Mawsters/ Nyan Maw Htun	Included diagrams, finalised document
1.3	27/03/2024	Mawsters/ Kenneth, Maw Htun	Officialised new project name, shelvd

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Shelvd is an advanced web-based platform designed to revolutionize the way users interact with literary content. This Release Plan document outlines the strategy and considerations involved in rolling out the system in a phased approach to ensure optimal user experience and system performance.

2. REFERENCED DOCUMENTS

This Release Plan is in alignment with the Software Development Plan, Project Management Plan, and the Communication Management Plan. These documents collectively outline the broader strategic approach for the lifecycle of the shelvd project.

Table 1: Referenced Documents

Document Name	Document Identifier	Issuance Date
Software Requirements Specification	SRS	2024-02-21
Project Plan	PP	2024-02-21

3. OVERVIEW

Shelve will serve as an interactive hub for book enthusiasts, offering advanced search capabilities, personalized recommendations, and the ability to manage personal book collections. The platform will integrate with external APIs to ensure a comprehensive and up-to-date catalog.

4. ASSUMPTIONS, CONSTRAINTS, RISKS

4.1. Assumptions

The user base will grow incrementally, allowing for scalability adjustments.

External API sources will remain available and stable throughout the release cycle.

4.2. Constraints

The project is bound by a limited initial budget, necessitating a lean initial release with core features that may be more rudimentary in its implementation, but remain highly practical in its usage (e.g. a recommendation system reliant on simple extraction of user’s preferred genres, rather than a complex implementation of artificial intelligence-based algorithms).

The timeframe for initial deployment is fixed, which may limit the scope of the first release.

4.3. Risks

A potential risk involves integration dependencies on third-party APIs, which could affect the release schedule.

User adoption rates may not meet projections, impacting subsequent release funding.

5. RELEASE APPROACH

5.1. Rationale

The release approach for shelvd is informed by the specifications outlined in the Software Requirements Specification document. These requirements serve as the foundation for determining the scope and priorities of the release. Assumptions regarding user growth and external APIs, as outlined in the Project Plan, influence our approach to scalability and integration strategies. Additionally, project constraints, including a limited budget and fixed deployment timeframe from the Project Plan, necessitate a phased approach prioritizing core functionalities.

5.2. Release Strategy

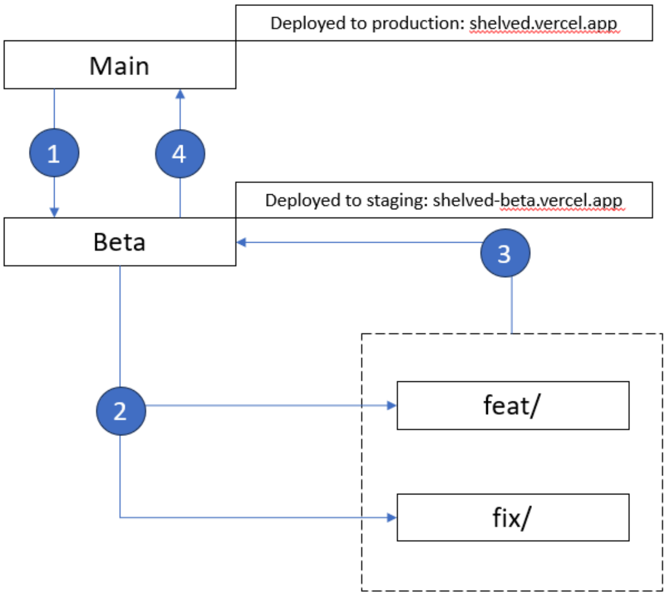
Deployed to production: shelved.vercel.app
Main
1
4
Deployed to staging: shelved-beta.vercel.app
Beta
3
2
feat/
fix/

Figure 1: Release Process Overview

A combination of functional and user base rollouts will be employed to introduce features to segments of users progressively, primary segregated into production level and staging (beta) level. This approach ensures efficient delivery of core functionalities while allowing targeted testing and feedback collection. The goal is to optimize user adoption and value delivery throughout the release cycle. The figure details the release strategy management process:

- **Beta Branch from Main (1):**

  Initial developments are carried out within the beta branch, pre-release.

- **Feature Branching (2):**

  New features (feat/) and bug fixes (fix/) are developed in separate branches off the main codebase. This allows developers to work on changes independently without affecting the stable version of the software.

- **Feature Merging (3):**

  Once a feature or fix is completed, the changes are merged into a beta branch. This is a pre-production or staging branch where the code is deployed to a staging environment, on shelvd-beta.vercel.app. Here, more extensive testing and user acceptance testing (UAT) can be performed.

- **Beta Merging into Main (4):**

  After the changes in the beta branch have been thoroughly tested and are considered stable, they are merged into the main branch. The main branch represents the production-ready state of the codebase.

  Code from the main branch is deployed to the production environment deployed onto shelvd.vercel.app. This is the live application that users interact with.

- **Iterative Process:**

  The process is iterative, as indicated by the arrows. After a production release, new features and fixes continue to be developed, and the cycle repeats.

5.2.1. Release Content

Table 2: Release Content

Release	Content
Beta 0.0	Initialise baseline/boilerplate
Beta 1.0	Feature addition: API Implementation with server
Beta 1.1	Feature addition: ClerktJS (User Management)
Beta 1.2	Feature addition: Book Collection Management
Beta 1.3	Bug Fixes
Main 1.0	Minimum Viable Product
Beta 2.0	Error Fixes
Main 2.0	Main release

5.2.2. Release Schedule

Table 3: Release Schedule

Release	Projected Dates
Beta 0.0	08/02/2024
Beta 1.0	14/03/2024
Bea 1.1	21/03/2024
Beta 1.2	26/03/2024
Beta 1.3	27/03/2024
Main 1.0	27/03/2024
Beta 2.0	29/03/2024
Main 2.0	30/03/2024

5.2.3. Release Impacts

Table 4: Release Impact

Release	Content
Beta 0.0	The initial setup will lay down the essential architecture of the system, impacting the development process by establishing coding standards and repository structures. This will set the pace for subsequent features and releases.
Beta 1.0	The integration of the API with the server is a significant milestone that will begin to shape the end-user experience. It will necessitate updates to the business processes related to data management and may require additional training for staff to manage these new workflows.
Bea 1.1	The addition of ClientJS for user management will impact the registration and login processes. It introduces a new layer of interaction for users, possibly requiring updates to user guides to ensure smooth adaptation.
Beta 1.2	The book collection management feature will directly impact how users interact with the system, allowing them to organize and manage their reading material. This release will likely require updates to user guides and may involve additional help desk support to assist users in leveraging the new feature.
Beta 1.3	Focused on enhancing system stability and performance, bug fixes will improve user satisfaction and trust. It will have minimal impact on business processes but will significantly benefit the overall user experience.
Main 1.0	The minimum viable product (MVP) release will be the system's first exposure to a broader audience. This will significantly impact business processes as users begin to utilize the system in real-world scenarios. Training, support, and promotional materials will need to be updated and distributed.
Beta 2.0	This iteration focuses on refining the system post-MVP deployment, based on user feedback and identified issues. It aims to improve reliability and performance, directly benefiting user interaction and confidence in the system.
Main 2.0	This release marks the deployment of a stable, feature-complete version to all users. It will significantly impact business processes as it includes enhancements based on real-world use. This release will require

	comprehensive user training and may involve changes to existing protocols and procedures to accommodate new system features and capabilities.

In each release, the key objective is to incrementally deliver a more refined and user-friendly system. With each iteration, the emphasis is on addressing user needs, enhancing system robustness, and ensuring that business processes are streamlined to accommodate the new features and improvements. Stakeholders and end users will be kept informed of new capabilities, changes, and enhancements through timely and clear communication, supported by updated documentation and available training resources.

5.2.4. Release Notification

Stakeholders will be notified through email updates and webinars in the weeks preceding each release.

In addition to email updates and webinars, stakeholders can stay informed about the latest developments by following updates on the project's GitHub repository and the Wiki page maintained by Team Mawsters. The repository will include detailed commit histories, and the Wiki will offer comprehensive documentation and release notes.

The notification methods will be as follows:

Table 5: Release Methods

Stakeholder	Notification Method
End Users	Email, Webinars
Developers	GitHub, Wiki
Contributors (Team Mawsters)	Email, GitHub

Updates to the GitHub repository will be made regularly as new features and fixes are committed. The Wiki will be updated in tandem to reflect new documentation to assist with the transition to new releases.

6. GLOSSARY

Shelved

Shelved is the latest and official software project title and website title. Shelvd is the initial project working title prior to 14 March 2024. Both refer to the same project by Team Mawsters.

Beta

Pre-release software version for testing by select users, subject to further refinement.

Functional Rollout

Deploying specific software features in stages or phases to users.

Main

Stable software version for general release to all users, rigorously tested before deployment.

Phased Rollout

Deploying software to users gradually in stages or phases.

User Base Rollout

Deploying software to user segments in stages for testing and adoption.

Minimum Viable Product

A minimum viable product is a version of a product with just enough features to be usable by early customers who can then provide feedback for future product development.