**Brionna Morris**

**& Angel Jackson**

**Present**

**SHELVD**

**PROJECT PLAN**

**TABLE OF CONTENTS**

INTRODUCTION……………………………………………………………………….…….…2

PROJECT ORGANIZATION……………………………………………………………….…2

RISK ANALYSIS……………………………………………………………………………….…3-4

HARDWARE & SOFTWARE REQUIREMENTS……………………………….…….5-6

WORK BREAKDOWN STRUCTURE…………………………………………….………7

PROCESS FLOW DIAGRAMS………………………………………………………..……8

PROJECT SCHEDULE…………………………………………………………………………9

MIONITORING & REPORTING MECHANISMS……………………………………10

APPENDIX…………………………………………………………………………………..……11

**INTRODUCTION**

This project, called SHELVD, will provide an alternative solution for readers that like keeping track of the books they read. The project will provide users with a web application that allows them to store their books similar to Letterboxd or Good Reads. Users also can rank their books, add to a “like” category, provide a rating on the book and create an optional review of the book. The web application will allow users to add any book of any genre to their list. In addition, to adding a book to their library shelf, they will be able to select the book, as it will be a part of a larger database for the user to choose from.

**PROJECT ORGANIZATION**

|  |  |  |
| --- | --- | --- |
| Position | Names | Description |
| ***Project Manager*** | Brionna Morris | Ensures that the project is executed on time and within scope. Keeps track of all materials, organizes meetings with group, and monitors project. |
| ***Designer*** | Brionna Morris & Angel Jackson | Ensures that the project is feasible and can be executed in the allotted time assigned for the project. |
| ***Programmer*** | Angel Jackson | Writes the code for the book storage/tracking system for users. |
| ***Tester*** | Angel Jackson | Analyzes the program and makes sure that the code/program is bug free, and is flawless in design and functionality for future users. |

**RISK ANALYSIS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RISK** | **DESCRIPTION** | **RISK PRIORITY** | **RISK PLANNING** | |
| The web application cannot be developed, tested, and sent to product owner by project due date. | -The full development of the application is beyond the expertise and resource availability of current team, so the team will create a smaller beta version of the full application. | **HIGH RISK** | Implementing the agile methodology throughout the project, the team will adhere to the project schedule to complete the project on time. |
| The product may need approval to use images, titles, and likeness of intellectual property. | -The product sponsor and stakeholders will need to consult legal.  -Development team will pursue closed beta version | **HIGH RISK** | Since the web application will be released to stakeholders and the product owner in a beta form, they will be able to access images, titles, etc., but will need to consult legal prior to web application release. |
| The web application has strong app competition. | -Current team is pursuing unique positioning for application.  -Other teams may have resources to build larger application. | **MEDIUM RISK** | While this is not a huge concern, as there are many competing web applications on the market, Shelvd’s unique book integration with user experience focus potentially provides an edge over competitors. |
| Database might not contain every book from every genre necessary for user to add to library. | -Database storage will be limited.  -Team is too small and lacks expertise and resources to establish database large enough for all genres and titles. | **MEDIUM RISK** | Since this is a beta app there will be a select few titles, mainly NY Times Bestseller titles added to the database. In the future, there is a potential to add other titles not on the list, but self-published titles will remain absent from the database. |
| The user might not like the web application interface | -Each user may have different expectations in terms of design.  -Team will focus on a generalized user experience, design, and functionality. | **LOW RISK** | The design and development team will focus on user function in a way that makes the web application easy to use, but the team cannot control user likeability. |

**Cost Risk**: To maintain control of finances, a pseudo-budget will be developed for the full scope of the project to determine cost and ability to operate within project management constraints.

**Technology Risk**: The team has built in an extra week during the development phase in case there are issues with technology. If there is an issue, there are public educational resources that will be consulted to complete the project.

**Scope Creek Risk**: The group has built in an extra week in case scope creep impacts the project, however, the current plan to manage this risk is to scale back on aspects of the project to compensate for potential time issues.

**Skills Resource Risk**: Currently the group is pooling their knowledge resources to ensure the success of the project but have already discussed additional educational resources that will be consulted to ensure the success of the project.

|  |  |
| --- | --- |
| RISK | DESCRIPTION |
| Late Delivery | The full development of the application is beyond the expertise and resource availability of current team, so the team will create a smaller beta version of the full application. |
| Copyright Infringement | -The product sponsor and stakeholders will need to consult legal.  -Development team will pursue closed beta version |
| Web Application Competition | Users might find other applications better. |
| Database Incompatibility | The database might be too small for the full scope of the project. |
| User Experience | The application might to be sufficient for users to understand or add books to their library. |
| Backend Failures | Issues with code might arise that impact functionality of project. |
| Testing Failures | Debugging and testing current project might not be sufficient. |
| Team Availability & Scheduling Conflicts | Possibility of development team missing days or scheduled time off. |
| Scope of Project | Project might be too large with current resource availability. |
| Limited Understanding of Development Team | Project might require advanced understanding that current development team might not have. |
| Project Idea Changes | Due to scope of project, ideas may have to be modified to meet deadline. |
| Project Presentation | There might be an issue with overall project that impacts the presentation of project. |

**HARDWARE & SOFTWARE REQUIREMENTS**

**Hardware**

Given that this is a web-based application, the hardware requirements are primarily focused on the server-side components needed to host the web application and the client-side components that users will use to access the application. The server-side components are crucial for hosting the web application and must be robust and scalable, with sufficient storage, memory, and processing power to accommodate multiple concurrent users and the database's storage needs. A high-speed and reliable internet connection is also emphasized to ensure uninterrupted service to the users. On the client side, the focus is on the devices and browsers users will employ to access the application. This section underscores the importance of having adequate hardware infrastructure to ensure the seamless operation and accessibility of the SHELVD application, providing a foundation for the software components to function optimally.

**Server-side Components**

* **Server**: A robust and scalable server to host the web application. The server should have sufficient storage, memory, and processing power to handle multiple concurrent users and the database's storage needs.
* **Storage**: Adequate storage space to store the application's database, user data, and any other necessary files.
* **Network**: High-speed and reliable internet connection to ensure uninterrupted service to the users.

**Client-side Components**

* **Device**: Users will need a device with internet connectivity, such as a computer, tablet, or smartphone, to access the web application.
* **Browser**: A modern web browser that supports the latest web standards, such as Google Chrome, Mozilla Firefox, Safari, or Microsoft Edge.

**Software**

The software requirements for the SHELVD project are categorized into development, deployment, and user interface components. This section provides a comprehensive overview of the development, deployment, and user interface components essential for the realization of the SHELVD project. It details the programming languages, web frameworks, database management systems, and front-end technologies that will be employed to develop the application. Python is highlighted as the primary programming language for backend development, with the potential use of web frameworks like Flask or Django. The section also outlines the deployment components, including the web server, hosting platform, and more, ensuring the application's availability and security on the web. Additionally, the user interface components are specified, focusing on design tools and front-end frameworks to create a responsive and user-friendly interface.

**Development Components**

* **Programming Language**: Python will be the primary programming language used for developing the backend of the application.
* **Web Framework**: A suitable web framework like Flask or Django for developing the web application.
* **Database Management System**: A reliable DBMS like PostgreSQL or MySQL to manage the application's data efficiently.
* **Front-end Technologies**: HTML, CSS, and JavaScript will be used to create the user interface of the web application.
* **Version Control System**: A system like Git for tracking changes in the source code during development.

**Deployment Components**

* **Web Server**: A web server like Apache or Nginx to serve the web application.
* **Hosting Platform**: A hosting platform like AWS, Heroku, or DigitalOcean to deploy the application.
* **Domain Name**: A registered domain name for the web application to make it accessible via the web.
* **SSL Certificate**: An SSL certificate to secure the communication between the user's browser and the server.

**User Interface Components**

* **User Interface Design Tool**: Figma will be used for designing the user interface.
* **Front-end Framework**: A front-end framework like Bootstrap or Materialize may be used to aid in developing a responsive and user-friendly interface.

**WORK BREAKDOWN STRUCTURE**

PLANNING

TESTING

DATABASE

DEVELOPMENT

DESIGN

Backend

Blueprint

Book Titles

Shelvd Application

Testing

Book Posters

Testing

UI Design

Forms

Testing

Authors

Class

Database

Genres

Publisher

UML

Relationship

**PROCESS FLOW DIAGRAMS**

A graphical representation is provided, illustrating how information flows within the SHELVD system. This diagram shows the interactions between external entities, processes, data stores, and data flows, offering a visualization of the system's functionalities and data interactions. By decomposing the system into its fundamental components, the DFD aids in unraveling the complexities of the system, allowing for a clearer understanding and communication of how data is processed, stored, and exchanged within the system. This visual tool is helpful to analyzing and refining the system's design, ensuring alignment with the project's requirements and objectives. A diagram of a software system

Description automatically generated

**PROJECT SCHEDULE**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Activity | Oct.1 | Oct.8 | Oct. 15 | Oct.  22 | Oct. 29 | Nov.5 | Nov. 12 | Nov.19 | Nov. 26 | Dec.3 | Dec. 10 |
| Planning |  |  |  |  |  |  |  |  |  |  |  |
| Development |  |  |  |  |  |  |  |  |  |  |  |
| Database |  |  |  |  |  |  |  |  |  |  |  |
| Design |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |
| Launch |  |  |  |  |  |  |  |  |  |  |  |

**MONITORING & REPORTING MECHANISMS**

Monitoring and reporting mechanisms are crucial components in project management, ensuring that all team members are aligned, informed, and working cohesively towards the project goals. They facilitate effective communication, collaboration, and scheduling, which are pivotal for the successful execution of the project. For the SHELVD project, a combination of tools has been chosen to aid in these aspects, enhancing the overall workflow and productivity of the team.

**Communication and Collaboration: Slack**

Slack has been chosen as the primary tool for communication and collaboration within the team. It offers a centralized platform where team members can interact, discuss ideas, share updates, and address issues in real-time. The organized channel structure in Slack allows for topic-specific discussions, ensuring that conversations are focused and relevant. The integration capabilities of Slack also enable the team to connect other tools and services, streamlining the workflow and keeping all necessary information and notifications in one place.

**Scheduling: Calendars & Slack Updates**

To manage availability and schedule meetings, the team will share their availability through calendars and share any changes through Slack. This approach ensures that all team members are on the same page regarding each other's availability, allowing for efficient scheduling of meetings, discussions, and collaborative sessions.

**Design and Development Collaboration Tools: Figma, FigJam, Notion, and Trello**

As the team embarks on the design and development phases, additional collaboration tools will be considered to enhance the design workflow and project management. Possibilities include Figma and FigJam for collaborative design and prototyping, allowing team members to work together in real-time on design concepts and user interfaces. Notion could serve as a comprehensive workspace for documentation, task management, and knowledge sharing, ensuring that all project-related information is organized and accessible. Trello, with its intuitive board and card system, can aid in task tracking and project planning, providing a visual overview of the project’s progress and pending tasks. These offer free tiers and since a team member has premium access to some, additional useful features can be used at no additional cost.

**APPENDIX**

In the appendix, a detailed activity table is included, outlining the various tasks essential for the project, their estimated completion times, and dependencies. This table serves as a structured overview of the project's workflow, allowing for a clear understanding of the sequence and interdependencies of tasks. It encompasses activities ranging from initial steps like form and class creation to intricate tasks like backend development and final testing, ensuring a comprehensive representation of the project's developmental journey. The detailing in this section aids in foreseeing potential overlaps and bottlenecks, facilitating smoother execution and timely completion of the project.

| **Activity Number** | **Activity Description** | **Estimated Time** | **Dependencies** |
| --- | --- | --- | --- |
| **1** | **Forms** | **2 Days** | **None** |
| **2** | **Classes** | **3-4 Days** | **None** |
| **3** | **Database UML** | **1-2 Days** | **1-2** |
| **4** | **Database Relationships** | **1-3 Days** | **1-3** |
| **5** | **Backend Dev** | **Days** | **1-4** |
| **6** | **Dev Testing** | **2-3 Days** | **4** |
| **7** | **Book Titles** | **1 Day** | **None** |
| **8** | **Book Posters** | **1 Day** | **7, 9-11** |
| **9** | **Authors** | **1 Day** | **None** |
| **10** | **Genres** | **1 Day** | **7, 9** |
| **11** | **Publishers** | **1 Day** | **7, 9** |
| **12** | **UI Wireframe** | **2 Days** | **1-6** |
| **13** | **UI Design** | **3-4 Days** | **12** |
| **14** | **Testing UI** | **2 Days** | **13** |
| **15** | **Final Testing** | **1-3 Days** | **1-14** |