Project Initiation Document (PID)

|  |  |
| --- | --- |
| Project Number: | 1 |
| Project Name: | GitHub Repository Finder |
| Approved Date: | 30-06-2023 |
| Author: | SRIHARI MADDINENI |
| Version: |  |
| Strategic Goal ID: |  |
| Business Plan ID: |  |

Table of Contents

Introduction…………………………………………………………...3

Project Definition……………………………………………………..3

Problem Statement………………………………………………………..3

Proposed Solution……………………………………………...................4

Project Objectives………………………………………………………...4

Project Scope and Exclusions…………………………………………….5

Key Stakeholders…………………………………………………………5

Project Deliverables………………………………………………………6

Project Planning………………………………………………………6

Project Team……………………………………………………………...6

Project plan and Timeline………………………………………………...6

Resources…………………………………………………………………7

Development and User Interface…………………………………………8

Testing and Quality Assurance……………………………………….8

Test plan and test cases ………………………………………………….8

Quality Assurance………………………………………………………..9

Deployment………………………………………………………….10

Deployment plan………………………………………………………..10

Glossary……………………………………………………………..11

# Introduction

The GitHub Repository Finder project aims to develop a simple web-based application that allows users to search and retrieve GitHub repositories based on a specific username. By leveraging the GitHub API, users will be able to enter a username and get a list of repositories associated with that username.

# problem statement

2.1 Problem Statement

Finding and exploring GitHub repositories for a specific username can be time-consuming and challenging for users. The existing search functionality on GitHub may not provide a straightforward way to list all repositories for a particular user, leading to suboptimal user experience and manual effort to retrieve repository information..

# proposed solution

The GitHub Repository Finder will provide a user-friendly interface where users can input a username and retrieve a list of repositories associated with that username. The application will interact with the GitHub API to retrieve the repository data and present it in an organized and accessible manner.

# Project objectivies

The objectives of the GitHub Repository Finder project are as follows:Ensure the platform's user interface is intuitive and accessible to users of all technical backgrounds, enhancing the overall user experience.

# Project Scope

The scope of the project includes:

Developing a web-based application for searching and retrieving GitHub repositories based on a username.

Integrating with the GitHub API to retrieve repository data.

Implementing a user interface with a single textbox for entering the username and a list to display the repositories.

# project exclusions

Authentication and user account management.

Repository creation, modification, or deletion functionality.

# Key stakeholders

Users: Individuals or developers who want to search and retrieve GitHub repositories for a specific username.

Development Team: Software engineers, designers, and technical professionals responsible for developing and maintaining the application.

Project Managers: Oversee project coordination, communication, and successful delivery.

Business Owners/Organizations: Stakeholders who may have an interest in utilizing the application for their development or research purposes.

Quality Assurance/Testers: Ensure the functionality, usability, and reliability of the application through testing and identifying issues or bugs.

# project deliverablies

User-friendly web-based application for searching and retrieving GitHub repositories.

List of repositories associated with the entered username.

Integration with the GitHub API to retrieve repository data.

# project team

The project team will consist of the following roles:

Project Manager: Responsible for overall project coordination, communication, and timely delivery.

# project plain and timelines

The estimated timeline for completing this project is as follows:

Phase 1: Requirements Gathering and Design

Duration: 1 week

Phase 2: Development and Integration

Duration: 2 weeks

Phase 3: Testing and Quality Assurance

Duration: 1 week

Phase 4: Documentation and Deployment

Duration: 1 week

# Resources

Development Team: Software engineers, web developers, front-end developers, back-end developers, and user interface designers.

Project Manager: Responsible for overseeing the project, coordinating resources, managing timelines, and ensuring successful project completion.

GitHub API: The API provider that offers access to repository data on GitHub. Access to the API and documentation is crucial for integrating and retrieving repository information.

Hardware and Software: Development computers or laptops, development and production servers, development tools, frameworks, and libraries.

# User Interface

# Test plan and test cases

Clearly define the scope of testing and the objectives to be achieved. Specify the exact functionalities that will be tested and the platforms or devices on which the testing will be performed.

Test Environment:

Describe the environment in which the tests will be conducted, including operating systems, browsers, or devices.

Test Data:

Identify the test data required for the test scenarios, such as specific usernames and the expected list of repositories.

Test Scenarios:

Define various test scenarios that cover different search use cases and user interactions.

Functional Testing:

Execute functional tests to ensure the application works as expected, validating the retrieval of repositories based on the entered username.

# Quality assurance

Conduct functional testing to verify the correct functioning of the GitHub Repository Finder.

Validate the accuracy of retrieved repositories based on the entered username.

Ensure the user interface is intuitive and accessible for users of all technical backgrounds.

Conduct usability testing to gather feedback on the user experience and improve the application's design and usability.

# Deployment

Pre-Deployment Preparation:

* Finalize the production environment infrastructure, including hosting, servers, and domain configuration.
* Ensure that the necessary hardware, software, and network resources are available and properly configured.
* Verify that the required dependencies and libraries are installed on the production environment.

Deployment Package Preparation:

* Create a deployment package that includes all the necessary files, code, and configurations required for the application to run.
* Perform a final code review and ensure that all necessary files are included in the package.
* Create a version control tag or branch for the deployment package to track the deployed code.

Deployment Execution:

* Transfer the deployment package to the production environment.
* Execute deployment scripts or commands to deploy the application code and assets to the production server.
* Monitor the deployment process to ensure it completes successfully without any errors or interruptions.
* Verify that the deployed application is accessible and functioning as expected.

Monitoring and Logging Setup:

* Set up monitoring tools and configure performance monitoring, error tracking, and logging mechanisms.
* Monitor key metrics such as response time, resource utilization, and error rates to ensure the application's health and performance.
* Configure alerts and notifications to proactively detect and address any issues or anomalies.

Post-Deployment Tasks:

* Conduct post-deployment testing and validation to ensure the stability and correctness of the deployed application.
* Coordinate with the support and maintenance team to transition the application into their responsibility for ongoing support and maintenance.
* Update documentation and knowledge base with any specific deployment instructions or troubleshooting tips.

# Glossary

|  |  |
| --- | --- |
| Repository: | A collection of files, documentation, and version control history associated with a project on GitHub. |
| GitHub api | An application programming interface (API) provided by GitHub that allows developers to interact with GitHub's features and retrieve repository data. |