SRS-Bugtracker Project

1. Introduction

> 1.1 Purpose:

Ein Vollständiges, funktionierendes Bugtracking Tool Programmieren, das zumindest in der Lage ist, eine Datenbank von Bugs zu erstellen, mindestens 3 verschieden Zugriffslevels(Admin, Tester, Developer) kennt. Daraus folgt, dass es eine Möglichkeit geben muss, sich anzumelden und außerdem, als Admin Zugriffsrechte zu verwalten.

> 1.2 Document Conventions:

Im folgenden wird „Geomanist“ für Überschriften und „Calibri light“ für Texte verwendet. Texte sind außerdem eingerückt und Unterüberschriften unterstrichen. Die jeweiligen Schriftgrößen sind: 18pt Ü, 14pt UÜ, 12pt Texte.

> 1.3 Intended Audience and Reading Suggestions:

Developer: Als Anleitung zur Erstellung des Programms und später als Nachschlagewerk für verschiedene Funktionen.

Tester: Als Bedienungsanleitung. Hier sind vor allem die Abschnitte 2 und 4 interessant.

Sonstige: Als Information über das Projekt selbst und das Zustandekommen.

> 1.4 Product Scope:

Das Bugtracking Tool soll dabei helfen, das bearbeiten von Bugs, oder anderen (auch nicht Softwaremäßigen) Problemen zu erleichtern und vor allem übersichtlicher zu gestalten. Das soll eine deutliche Zeitersparnis mit sich bringen bzw die Abhängigkeit von dritten reduzieren, falls schon ein vergleichbares Tool in einzelnen Bereichen verwendet wird. Es soll außerdem durch Übersichten Vorgesetzten die Möglichkeit bieten, Arbeitsabläufe und die Arbeitsaufteilung zu verbessern.

> 1.5 References:

Link zum Projekt:

SRS: https://github.com/rick4470/IEEE-SRS-Tempate

1. Overall Description

> 2.1 Product Perspective:

Das Tool ist neu und wird kein altes ablösen, es soll unabhängig von anderen Programmen arbeiten, aber in einem späterem Entwicklungsschritt automatisiert gewisse Bugs aus Fehlerberichten anderer Programme lesen können.

> 2.2 Product Functions:

\* Das Tool muss eine grobe Übersicht über alle offenen Bugs bieten können.

\* Es muss die Möglichkeit bieten, User mit verschiedenen Rechten zu erstellen.

\* Es muss eine einfache und intuitive Möglichkeit bieten, neue Tickets zu erstellen.

\* Es muss dem User die Möglichkeit einer Suchfunktion bieten

\* Es muss die Möglichkeit bieten, bereits erstellte Tickets zu kommentieren und zu bearbeiten

\* Es muss die Möglichkeit bieten, neue Projekte zu erstellen

> 2.3 User Classes and Characteristics:

Manager: Muss nur Übersicht über Tickets und Projekte sehen, um die Mitarbeiter kontrollieren zu können

Admin: Kann neue User anlegen, User löschen Passwörter zurücksetzen und Berechtigungen vergeben.

Erfahrener Developer: Braucht eine schnelle Übersicht des Problems und die Möglichkeit Tickets zu bearbeiten, falls die ursprüngliche Beschreibung des Problems / ToDo’s nicht genau passt.

Junger Developer: Braucht eine schnelle Übersicht und Platz, sich Notizen zu machen.

Tester: Braucht die Möglichkeit, schnell und unkompliziert ein neues Ticket eröffnen oder verändern zu können, aber auch Screenshots o.ä. anhängen zu können.

Projektmanager: Muss neue Projekte anlegen, Tickets erstellen und Dringlichkeiten festlegen können. Sollte außerdem auch auf nach Projekten aufgeschlüsselten Übersichten zugreifen können.

> 2.4 Operating Enviroment:

Das Tool soll Webbasiert sein, und zumindest über Chrome und Firefox funktionieren. Die Software selbst soll auf einem Linux Server laufen.

> 2.5 Design and Implementation Constraints:

Das Frontend ist in HTML, CSS und JavaScript zu halten und es ist möglichst auf Abhängigkeit von dritten zu verzichten (Ausfallssicherheit bei Internetproblemen).

Das Backend ist in PHP, Python oder Java zu halten und als Datenbank soll eine Kostenlose, die Datensicherheit garantierende, leicht skalierbare verwendet werden.

> 2.6 User Documentation:

Es ist eine Quickstart-Anleitung vorgesehen, ob interaktiv im Programm oder als PDF obliegt den Programmierern.

> 2.7 Assumptions and Dependencies:

Die Geschwindigkeit der Umsetzung könnte durch das Studium des Hauptentwicklers beeinträchtigt werden, oder durch das erlernen neuer Technologien beschleunigt werden.

Abhängigkeiten sollten bis auf Bootstrap und ein etwaiges Tool als Hilfe bei der Login Gestaltung keine bestehen.

1. External Interface Requirements

> 3.1 User Interfaces:

Das User Interface soll möglichst übersichtlich gestaltet werden. Es gibt auf der linken Seite ein Menü, mit dem man schnell auf die für den jeweiligen User wichtigsten Funktionen zugreifen kann. Oben gibt es neben der Überschrift und der Möglichkeit sich auszuloggen noch ein Seitenweites Suchfeld.

Die Login Seite ist möglichst einfach gestaltet, bietet aber zumindest die Zusätzlichen Funktionen, das Passwort zurückzusetzen und sich als Demo-User(mit Dummy-Daten) einzuloggen.

Generell werden wenige verschiedene Farben verwendet, um die Lesbarkeit zu erhöhen und Ablenkungen zu eliminieren.

> 3.2 Hardware Interfaces:

1. Other Nonfunctional Requirements

> 5.1 Performance Requirements:

Das System sollte auch ohne Internet funktionieren und es dürfen keine Tickets verloren gehen. Das System muss mit gleichzeitigen Anfragen umgehen können und darf keine merkbaren Verzögerungen verursachen.

> 5.2 Safety Requirements:

Der Zugriff von nicht in berechtigten Personen muss verhindert werden. Die Daten, die in der Demo-Version verwendet werden, dürfen keine Rückschlüsse auf echte Daten zulassen. Für die Passwörter gibt es gewisse einzuhaltende Bestimmungen. Außerdem muss die Datenintegrität zu jedem Zeitpunkt gewährleistet sein.

> 5.3 Security Requirements:

Wie schon erwähnt, muss die Datensicherheit gegeben sein und ein Zugriff durch externe mit allen möglichen Mitteln verhindert werden.

> 5.4 Software Quality Attributes:

Die Software muss in allen gängigen Browsern und dabei zumindest auf Laptops und Tablets einwandfrei laufen. Es sollte zu keinen Ausfällen oder durch die Software verursachten Fehlern auch bei sehr starker Nutzung des Systems kommen.

Jeglicher Code ist ausreichend zu kommentieren und übersichtlich zu gestalten, bzw einzelne Dokumente, die Code enthalten sind übersichtlich in Verzeichnissen usw anzuordnen. Insgesamt ist auf Wartbarkeit und Übersichtlichkeit zu achten.

> 5.5 Business Rules:

Ein Administrator kann Passwörter vergeben, einzelnen Usern Zusatzberechtigungen erteilen und neue User anlegen.

Ein Manager kann Übersichten verschiedener Art erstellen und abrufen und neue Projekte erstellen und diesen Projektmanager zuweisen.git