Project: Bug Tracker

Software Requirements Specifications

Version 3.0

Note: This template is largely based on the template provided by the Unified Process for Education (www.upedu.com) with minor modifications.

CODER FOUNDRY
1231 SHIELDS ROAD SUITE 5
KERNERSVILLE, NC 27284
PHONE: 336.231.8632

Revision History

Date	Version	Description	Author
08/14/14	1.0	Initial document	Coder Foundry
02/12/15	2.0	Updated requirements specifications	Coder Foundry
10/18/16	2.1	Updated Assumption software version	Antonio Raynor
03/04/2017	3.0	Requirement Updates and Edits	Antonio Raynor

Table of Contents

Section 1: Document Overview

This document is intended as a guide to Coder Foundry students for developing their midterm project, a bug tracking system employing MVC and SQL Server Database technologies. This document describes the complete project specification and includes requirements, sample screen mockups, and complete database design specifications.

Section 2 discusses the project's purpose, stakeholders, and any assumptions and constraints under which the student is expected to operate.

Section 3 identifies and discusses all project requirements (business, functional, and non-functional).

Section 4 provides classification and prioritization of the functional requirements.

Section 5 is the Appendices section, which may include screen mockups and/or database design parameters.

Section 2: Project Overview

Project Introduction

The purposes of all Coder Foundry student projects are two-fold: 1) to teach students relevant and marketable coding skills and best practices, and 2) to serve as audition pieces for prospective employers. As such, it is imperative that student projects be of a type that is immediately recognizable and relatable to prospective employers. If a prospective employer has to ask questions such as: "Why did they write this?" or "What is this used for?" of "How does this work?", then we have not adequately met the needs of our students.

The Bug Tracker is a Web-based software application designed to mirror any of a wide variety of bug tracking software systems in use by corporations throughout world. If a company develops software, whether for in-house use or distribution, it uses a bug tracking system to maintain an accurate record of the software development process on a per-project basis.

Stakeholders

This project has no direct client. The stakeholders in this case are the student and the Coder Foundry administrative staff, who will be responsible for grading the project as Pass/Fail.

Assumptions and Constraints

This project is to be built as a Web-based service application using the Microsoft Visual Studio 2015 IDE (any edition is acceptable, though we encourage the use of the 2015 Community edition, which is available free of charge), MVC, MS Code-First database schema using Entity Framework with user- and role-level security, and web pages built with HTML5, Twitter Bootstrap, and model binding with Razor. Student projects will be hosted online through Microsoft Azure Web hosting, with a link to each student's version of the project placed on the student's personal Website.

Section 3: Requirements

This section describes all requirements, both functional and non-functional, for this project as set forth by Coder Foundry staff and administration. All requirements outlined herein must be met and demonstrated in the resulting software application for the project to be considered "complete." Any missing requirements/features will result in the project being deemed inadequate.

Functional Requirements

End User Requirements

End users must be able to perform the following actions:

	Description	**Purpose**
1	Register as a user / Login	Secure system access
2	Assign/unassign users to/from roles	Role-based security
3	Create projects	Organization of resources / project managers or admin only
4	Assign/unassign users to/from projects	Organization of resources / project managers or admin only
5	Create tickets	Log software issue instance
6	Assign tickets to user	A developer must be responsible for a ticket item, enforce accountability
6	Edit submitted tickets	Make modifications to existing tickets
7	Create ticket comments	Add progress comments and other important information to the ticket history
8	Create ticket attachments	Add helpful visuals and other documentation to the ticket history
9	List comments, attachments per ticket	Organization of these resources
10	List history of a ticket's changes	Very important view for the developer on the project
11	List tickets by owner	Provide easy tracking of tickets logged by a particular user
12	List tickets by assignment	Provide easy tracking of tickets assigned to a particular developer
13	List ticket by project	Organization of resources – easy access to tickets per software project
14	Filter ticket lists by ticket type, priority, and status	Ease of use and access
15	Filter ticket lists by creation date/time (i.e. all tickets after indicated date/time)	Ease of use and access
16	Sort ticket list by title, owner, assignment, creation or recent update date/time, ticket type, priority, status, and project	Ease of use and access
17	Full text search of all relevant fields	Ease of use and access

User Interface Requirements

To facilitate intuitive and user-friendly interaction, the user interface must be designed according to the following requirements:

Req. #	Page Description	Feature List
1	Landing	1) All users to this page when accessing the site
		2) Links to Login and Register, include button to Login as Guest or Demo (this user should have Admin rights to the site for demonstration purposes)
2	Login / Register	1) From this page users can login, register as a new user, and recover a lost password
		2) Optional: Include Social Networks as a registration/login method
3	Dashboard	1) Users are directed here on successful login
		2) Display the most recent ticket assignments or tickets logged for the user (depends on role) or other relevant role-based information
		3) Optional: Display data visualization of ticket types logged and resolved per project
4	Role Management	1) Admin users must have access to an interface for assigning users to and removing users from the following roles (roles should be seeded in database along with global admin user): • Administrator • Project Manager • Developer • Submitter
5	Ticket Management	1) Paged listing of ticket items (default 10 tickets per page, may be changed by user)
		2) Users have access to ticket lists based on role: • Submitters see tickets which they own • Developers see tickets to which they are assigned • Project managers see tickets belonging to the projects for which they are responsible • Administrators see all tickets
		3) Tickets may be created and edited, not deleted. Completed tickets receive a “Resolved” status.
		4) A ticket details page must be used to provide full detailed ticket history for each ticket
		5) Ticket statuses, types, and priorities should be seeded in database
		6) Provide interfaces for adding comments and attachments
		7) Developers should be notified via email or other messaging service when assigned a new ticket

6	Project Management	1) Project managers and administrators must have access to an interface for creating projects
		2) Project managers and administrators must have access to an interface for assigning users to and removing users from a project
7	User Profile	1) Change password
		2) Edit profile information (name, email, etc.)
		3) Associate profile with social network identities

Non-functional Requirements

Non-function requirements are specific requirements, usually set forth by the client or by third-party product dependencies, which are not directly related to application features. These may include, but are not limited to, performance, security, and software quality requirements.

Performance Requirements

As a commercial-grade product, this project is expected to perform to professional standards. That is, its performance should be comparable with similar commercial products. Users should not be waiting impatiently for their Web pages to be delivered. Sorting and filtering of data should happen "instantaneously" based on user perception. Care should be taken to ensure the performance of the software enhances the user experience, rather than detracting from it.

Security Requirements

This project must be fully secured using Authentication and Authorization through Microsoft Entity Framework to prevent unauthorized persons from accessing user profile and account information. Stored Procedures or server-side LINQ statements, among other considerations, may be used to prevent SQL injection attacks.

Software Quality Attributes

This project is expected to appear and function as a comparable commercial-grade product. It is intended to serve as an audition piece to prospective employers, and therefore must adhere to commercial standards of software development, not just in function, as mentioned above, but in form and design as well.

Section 4: Classification of Functional Requirements

All software requirements are categorized according to their priority:

1) Essential
2) Desirable
3) Optional

End User Requirements

	Description	Priority	Dependencies	Difficulty
1	Register new user / Login	1		
2	Assign/unassign users to/from roles	1		
3	Create projects	1		
4	Assign/unassign users to/from projects	1		
5	Create tickets	1		
6	Assign tickets to user	1		
7	Edit submitted tickets	1		
8	Create ticket comments	1		
9	Create ticket attachments	1		
10	List comments, attachments per ticket	1		
11	List history of a ticket's changes	1		
12	List tickets by owner	1		
13	List tickets by assignment	1		
14	List ticket by project	1		
15	Filter / Sort ticket list by: • Title • Owner • Assignment • Creation date/time • Update date/time • Type • Priority • Status • Project	1		
16	Full text search of any and all relevant fields (title, description, project, type, priority, status, owner, assignment)	1		

User Interface Requirements

	Description	Priority	Req. Ref. #	Completed (date)
1	Landing page providing links to login/registration and to login as guest/demo	1		
2	Include Social Networks as a registration method	3		
3	Users are directed to a dashboard after login	2		
4	Change / recover password	1		
5	Edit profile information (name, email, etc.)	2		
6	Admin users must have access to an interface for assigning users to and removing users from the following roles (roles should be seeded in database along with global admin user): • Administrator • Project Manager • Developer • Submitter	1		
7	Project managers and administrators must have access to an interface for creating projects	1		
8	Project managers and administrators must have access to an interface for assigning users to and removing users from a project	1		
9	Display the most recent ticket assignments or tickets logged for the user (depends on role) or other relevant role-based information	2		
10	Display data visualization of ticket types logged and resolved per project and/or total	2		
11	Paged listing of ticket items (default 10 tickets per page, may be changed by user)	1		
12	Users have access to ticket lists based on role: • Submitters see tickets which they own • Developers see tickets to which they are assigned • Project managers see tickets for the projects for which they are responsible • Administrators see all tickets	1		
13	Tickets may be created and edited, not deleted. Completed tickets receive a “Resolved” status.	1		
14	A ticket details page must be used to provide full detailed ticket history for each ticket	1		
15	Ticket statuses, types, and priorities should be seeded in database	1		
16	Provide interfaces for adding comments and attachments	1		
17	Developers should be notified via email or other messaging service when assigned a new ticket	1		

Section 5: Developer Add-ons

Use this section to attach an ERD for your database design configuration, as well as and screen mockups you come up with for your system. It will help speed up the development process if you fully design your database and sketch out some ideas for your user interface design before you start coding. This is standard practice.

You can use SQL Server, Microsoft Visio, or any other UML design tool (including pencil and paper) to create your database ERD. Likewise, you may use a mockup tool of your choice (including pencil and paper, but Balsamiq Mockups (https://balsamiq.com/) is good) to begin designing your UI.

Add these items to your project spec to show your Project Manager upon review of your first deliverable.