devc++githubsrs

Software Engineering (Lovely Professional University)

1. Introduction :- The introduction of the software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to gather and analyze and give an in-depth insight of the complete DEV C++ by defining the problem statement in detail. Nevertheless, it also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The detailed requirements of the marvel DEV C++ are provided in this documents.

1.1 Purpose :- The purpose of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to consumers. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outlines concepts that may be developed later and document ideas that are being considered, but may be discard as the product develops.

In short, the purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality. Nonetheless, its helps any designer and developer to assist in software delivery life cycle (SDLC) processes.

1.2 Scope :- This project is intended for making use to compiling C and C++ for great experience.

Usually they provide lots of extensive features to developers to ease application developers life.

This SRS is also aimed at specifying requirements of software to developed but it can also be applied to assist in the selection of in the house and commercial software products. The standard can be used to create software requirements specifications directly or can be used as a model for defining a organization or project specific standard. It does not identify any specific method, nomenclature or tool for preparing an SRS.

1.3 Overview :- We are going to focus on describing the system in terms of product perspective, product functions, user characteristics, assumptions and dependencies on the following section of this document. Next, we will address specific requirements of the system, which will enclose external interface requirements, requirements of the system, performance requirements, and other requirements.

2. Overall Description :- This section gives background information about specific requirements of the Dev C++ in brief. Although we will not describe every requirement in detail, this section will describe the factors that affect the final product.

2.1 Product perspective :- This software product is eventually intended for the software developers. Product will be deployed to web site and all users of the product will access by use of the website. Website will be main user interface where users can operate will the provided functionality. However, this web site will be only a part of a larger system. There will be cloud server where all the user data is kept and all the execution is done.

2.2 Product Functions :- This product, Dev C++, must have number of features which will allow user to use functionalities which have been explained above. Required functionalities of the product can be summarized in five categories, user management requirements, code editor requirements, debugger requirements, command line interface requirements and interface requirements. Over all description of the requirements can be found below.

2.2.1 User Management Requirement :- The category of requirements is related to user authentication mechanism and workspace management of users. Each user will have credentials to connect their workspace on cloud and will be assigned to workspace. Users will perform all the functionality over this workspace using his credentials.

2.2.2 Code Editor Requirements :- One of the most important functionality expected from an integrated development environment is a code editor which will ease the developer's life. Code editor will be the main interface that developers deal with. It supports variety of programming language with highlighting, syntax checking, auto indentation and language specific auto-complete.

2.2.3 Debugger Requirements :- Debugger is the main tool that developers can test and debug their target program. Debugger of the product should allow setting and displaying break points on the code. It will also provide functionality of stopping/continuing of the execution of debugger. Finally, it will provide an expression interface where user can enter an expression at each step.

2.2.4 Terminal Requirements :- As an important part of the software Development process, an integerated develop-

enivornment should provide a command line interface where user can work in old fashion and accomplish complicated tasks such as configuring git synchronization. Main component of CLI will be the terminal. Terminal will allow user to run UNIX command on his own workspace and also run predifined programs such as mvn, svn etc. Terminal will also provided auto-complete by list of available commands and browse in this command history.

2.2.5 Interface Requirements :- This group of requirements is related to external interaction of the workspace with outer world. For user to interact with the workspace product will provide both command line interface and graphical interface. Command line interface will be UNIX like and graphical interface will allow tabbed navigation of windows, hierarchical view of workspace etc.

2.3 User Characteristics :- Users of the Dev C++ will mainly be software developers. Since it is reasonable to assume that an average developer has knowledge about functionalities and usage of IDE, we assume that our users will already be informed about basic functionality of the product. Also clear documentation and tutorials about the product feature will be provided.

3. Specific Requirements :- With this section and later, I will describe the requirements of the software in detail. Basically, we will categorized requirements in three which are namely external interface requirements, functional requirements and non functional requirements.

3.1. External Interface Requirements :- In this sub section, we will describe the external interface requirements of the product in three categories which are

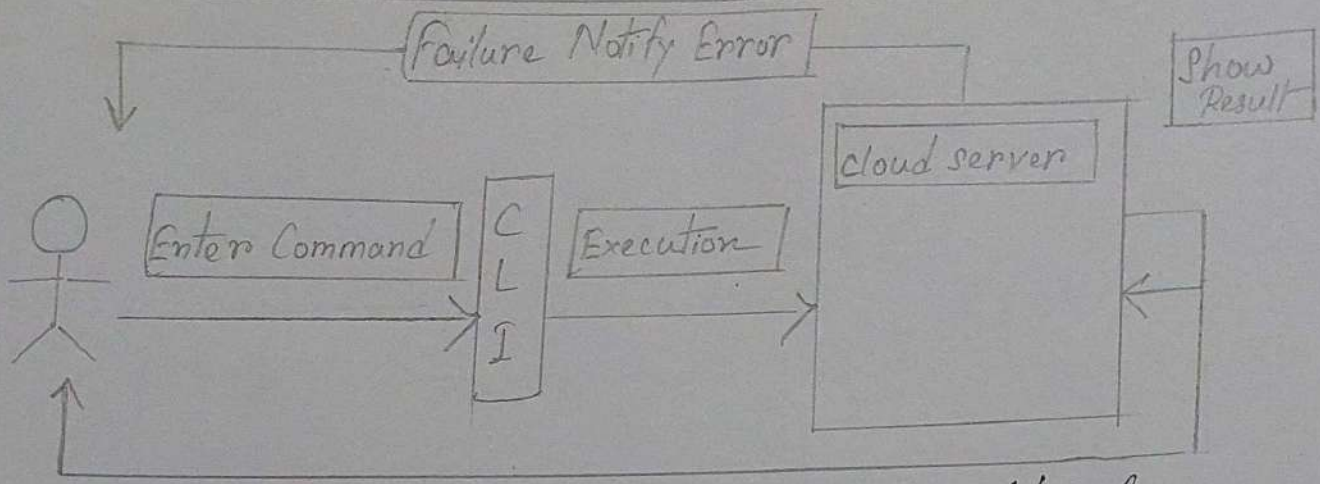
CLI support, workspace explorer and code editor and communication interfaces. The purpose of this section is to identify and document this interfaces and interaction of the software with external entities in detail.

3.1.1 External services Synchronization Interface :-

Product is going to provide an interface to publish the content of the workspace to external system. For the inital version of the product this interface works well with the github website. If user also a github account and wants to synchronization his repository with the github, product should be able to provide his functionality.

3.1.2 Command line Support :- To use the command line interface, user first should login to the system with his own credentials. After login user will be redirected to the main page. In the main page user can go to the below part of the page which will be command line where user can execute UNIX commands on the current directory.

During any process, user should be able to receive feedback about submission. If the command or program runs successfully, then the user should be able to see the effects either on the CLI or the in the workspace explorer. If an error occurs because of the user limitations, user should be able to given a detailed notification about the cause of the error and solutions of possible.



- Above diagram presents the functionality of Command Line Interface

3.1.3 Project/Workspace Explorer:- Project/workspace Explorer is going to be the main interface of the Dev C++. User will be able to their file hierarchy and edit it.

To able to use this main graphical user interface, users should have login to the system. After login, user will be directed to a main page specialized for himself. In this main interface there will be a workspace explorer where users can his see his file hierarchy and create and edit the files in the workspace.

There is also be a code editor in this main interface, so that users can be edit their source codes. Also they can use the command line interface to execute commands on the workspace which is also in the main interface of the system.

During all these operations, user should be notified about the status of the operation. If an operation operates successfully user should get a message or directly see the effect on workspace explorer or on the code editor. If an error occurs, users should get an error message indication the cause of error.

DEV C++

File | Project

Helloworld.cpp | terminal

Couses
- Cou1.cpp
- Cou2.cpp

```
1.
2.  using namespace std;
3.
4.  int main()
5.  {
6.      int test_variable=0, test=variable2=3;
7.      cout<<"Hello World";
8.      return test_1;
9.  }
10.
```

Interface

3.2 System Features :- In this subsection I will examine the features of the system in detail by categorizing them according to their functionality. For each of the feature, we will give an introduction, purpose, diagram and a stimulus/response sequence.

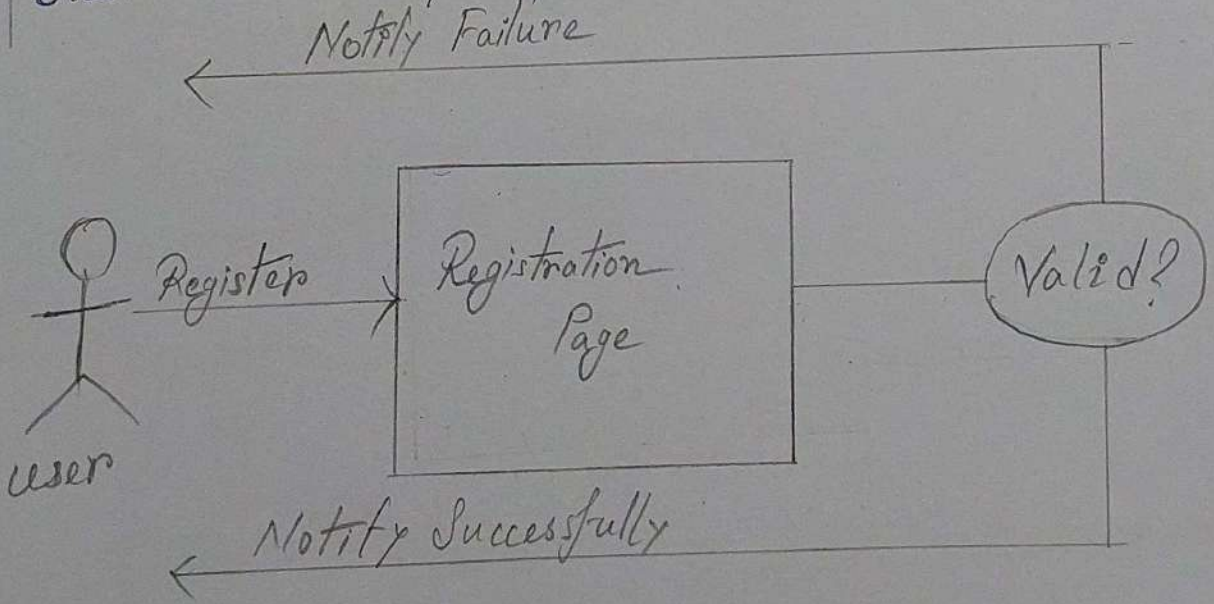
3.2.1 User Management

3.2.1.1 User Authentication

3.2.1.1.1 Background Information

User will have his own workspace, and he must be logged in to the server to access his workspace. Hence, first-time users must be complete registration process. To register to the system, user must specify some information asked during registration. After validation, registration will completed, and user will be informed.

3.2.1.1.2 Stimulus/Response Sequences



Description

Primary Actor	User
Goal in context	Purpose of this feature is to register user to the system
Triggers	User wants to register to the system

Normal Flow of Events

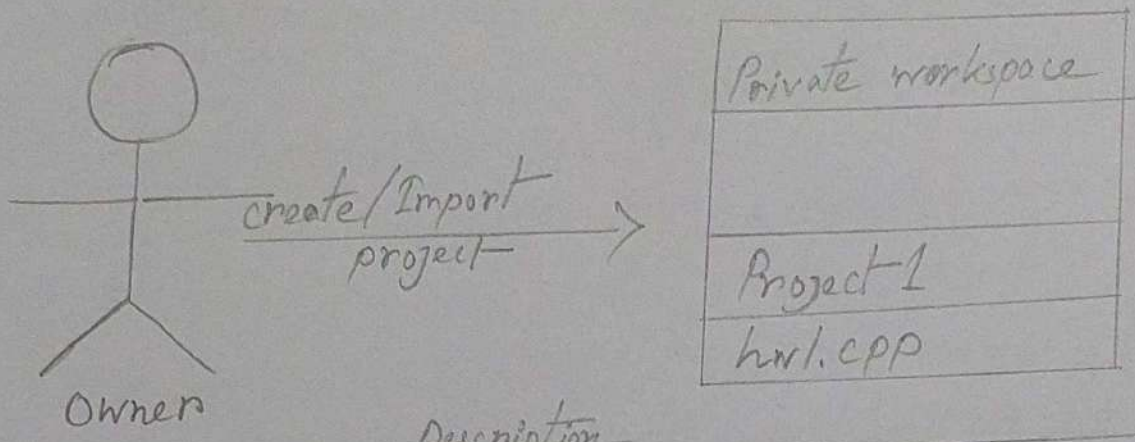
i) User opens the registration page.
ii) User specifies his information.
iii) System validate the specified information.
iv) User is registered to the system.

3.2.1.2 Workspace Management/Ownership:-

3.2.1.2.1 Background Information:- A workspace is a logical collection of projects. User can manage his projects within a workspace. User can create new projects, import existing projects, or create and delete files and folders in any of the project at any time.

After registration of users, a private workspace will be available for him. Workspace will be accessed by workspace owner with full access. Owner can change visibility of his workspace. If workspace is set to be as public, any user can observe his projects.

3.2.1.2.2 Stimulus / Response Sequences



Description	
Primary Actor	User workspace Owner
Goal in Context	Purpose of this feature is to create a new project or import existing one.
Preconditions	User must be logged into the system
Trigger	User wants to create a new project, or import existing project

Normal Flow of Events

1. User logins in to the system
2. User opens his workspace.
3. User creates a project.
4. User creates a file into the new project.

Functional Requirements

REQ1: The system shall provide a registration page.
REQ2: The system shall provide a login page.
REQ3: The system shall support creating and importing projects.
REQ4: The system shall support creation files or folders.
REQ5: The system shall support public workspace option.
REQ6: The system shall support github synchronization.