

# CHAIN REACTION

A fun game in c++

## *SRS DOCUMENT*

PROJECT BY:

*AAKASH PRALIYA*

*RATANJOT SINGH*

*ASHWIN RAI*

*SATYENDRAS. RAJPOOT*



# CONTENTS

## 1. Introduction

- I. Purpose
- II. Scope
- III. References

## 2. Overall description

- I. Product function
- II. Operation environment
- III. User characteristics
- IV. Design goals
- V. Design and implementation constraints
- VI. Assumptions and dependences

## 3. External Interfaces requirements

- I. User interfaces
- II. Hardware interfaces
- III. Software interfaces

## 4. Functional Requirements

## 5. Non Functional Requirements

## 6. Product Environment

## 7. Future Prospects

# INTRODUCTION

## I. Purpose

The purpose of this document is to describe the project of chain reaction game made using C++ language. This document contains functional and nonfunctional requirements of project and it also contains the guidelines to start over the project. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

The intended users of game/project are our honorable teachers, TAs and RAs and others related to CS101.

## II. Scope

- The game is specifically designed for the use of teenagers (more specifically freshers of IITB) and others. It is basically a fun oriented game which can run on any computer with most basic configuration of hardware and required software (SDL in code blocks). The game rules are easy to understand. We hope that the freshers can play and enjoy this project/game after some boring and creepy hours of study .Sleepy hours of lectures can be enjoyed using this game.

### III. References

- Wikipedia
- Google Play Store
- Lazy Foo SDL tutorials

# OVERALL DESCRIPTION

Chain reaction is a project for fun by students of introductory course on programming by using C++ language.

## I. Product Functions

Once started the game will provide user with three options:

- a. Play
- b. Instructions
- c. Quit

User can choose appropriate option. The game allows a maximum of four users to play a single game starting from single player. Colors of orbs are pre-decided. Users can take their appropriate colors by shifting their order of chances. The game takes input from keyboard when started. Once game is started it takes input directly from mouse and provides a graphical gaming experience. It has a special feature of artificial intelligence i.e. the game can be played against computer.

## II. Operating Environment

OS specification:

- a. Windows
- b. Linux

### III. User Characteristics

Players of this game are assumed to have basic knowledge of handling computer, presumption that players know the rules of the game.

### IV. Design goals:-

- a. Reliability: Game wouldn't crash with bad input.
- b. Usability/Usage continuity interaction: Choices for the players to make are :
  - 1 Player Game
  - 2 Player game
  - 3 Player game
  - 4 Player game
- c. Availability: Game can be played as long as certain conditions are met.

### V. Design and implementation constraints

The only major constraint to consider in the design of this project is within the software being used for the game's development. Since this software will rely on some game engine code, then any constraint on development will be based upon the boundaries of the game engine. The product shall run on desktop PC's and laptops.

## VI. Assumptions and dependences

- The application is not widely used as it is a project given to students of CS101 therefore it uses English as its standard language.
- The application will run on Windows XP/Vista/7/8 and Linux platforms with the C++ Runtime Environment installed.
- The application does not require an internet connection.

## External Interface Requirements

### I. User interface

A raw window takes input from user and game responds graphically. The graphic library “SDL/SDL.h” is used to apply graphics to the game. The key controls are mentioned in user manual. The game interacts with the user graphically. Window does not require any carriage return after the inputs and the keyboard characters are read.

### II. Hardware interface

Only the recommended configuration (basic requirements) of a computer system.

Minimal requirements include mouse, keyboard, monitor, and compiler to compile C++ code with special libraries (SDL). No other specific hardware is required.

### III. Software Interface

Operating system (Windows XP, Windows Vista, Windows 7, and Linux) and compiler to compile .cpp file.

# Functional Requirements

## I. Starting the game

1. Game starts upon clicking the executable equivalent file of the game.
2. Main Menu appears with the following options:-
  - a. PLAY
  - b. INSTRUCTIONS
  - c. QUIT

## II. In Game

- a. Mouse keys are used to interact with the game. Different players have different color of orbs (orbitals).
- b. Players have the freedom to choose their order of chances.
- c. Colors of orbs are pre-decided according to the order of chances.

## III. Want to play again!!!

If you are enjoying the game and want to play it again, you can play it again by pressing enter key after finishing of a game.

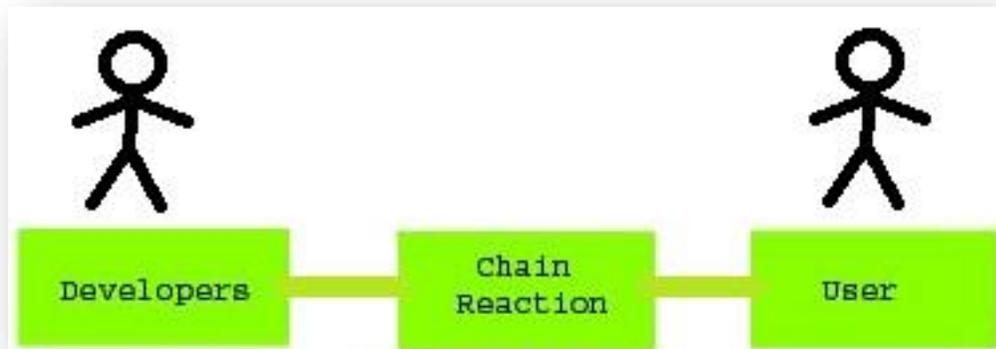
## IV. Exiting/Ending the Game

Player can press escape button to exit out of the game after the game is finished.

# Non-Functional Requirements

1. General Guidelines: Priority should be given to performance.
2. Operating Constraints: This program requires the SDL library to be present on the target machine and for the computer to have C++ Redistributable package in case of Windows and C++ Runtime Environment in case of Linux.
3. Documentation: The User Manual will be made for the application in addition to the INSTRUCTIONS included inside the 'Main Menu' of the game.
4. Portability: The program will run on Windows XP/Vista/7, and Ubuntu 10.04/11.04/11.10.
5. Reliability: As the game is for pure recreation and involves no user data, reliability is of low importance.
6. Deployment: The game will be uploaded online in the form of a C++ executable file.

## End Product Environment



The end product has two active actors. The main active actor in this system will of course be the user of this end product. But even the developers will remain active in the sense that they will help out in sorting out errors that may crop up and updates that may be required even after the project has been completed.

## Future Prospects

The game is simple interesting and it can have a good future if it will be used in mobile and tablets. Therefore we expect the game to be in other operating systems.

Although we worked hard for the project, certain conditions cannot be met as we have certain constraints of knowledge and time.

Yet, we hope that the users of this project certainly help us by checking the errors in game and upgrading it.