

**Simulation of Different Dead lock Scenarios and Avoiding the Deadlocks**

**Low-Level Design**

**Document Control:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | |
|  |  |  | |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | |
|  |  |  |  | | | |
|  |  |  |  | | | |

|  |
| --- |
| **Team Members** |

|  |  |
| --- | --- |
| **Employee ID:** | **Name** |
| 46282103 | Saranya Munigolla |
| 46282105 | Shwetha Julappayyanapalya Parashuramareddy |
| 46290161 | Geethanjali Sambana |
| 46290160 | Lakshmi Durga Pippara |
| 46282106 | Senthamil Selvi |
| 46290166 | Yelavarthi Sravya |

|  |
| --- |
| **Table of Contents** |

**I) Low-Level Design**

|  |  |
| --- | --- |
| 1. Introduction |  |
| 1.1 Purpose | 4 |
| 1.2 Document Conventions | 4 |
| 1.3 Intended Audience and Reading Suggestions | 4 |
| 1.4 References | 4 |
| 1. Detailed system design | 5 |
| 2.1 Design Description | 5 |
| 2.2 Flowchart | 6 |
| 2.3 Modules | 7 |
| 2.4 Use Case Diagram | 15 |

|  |
| --- |
| **Low-Level Design** |

**1. Introduction**

The aim of this document is to gather, analyze and give an in-depth insight into the complete simulation of different dead lock scenarios and avoiding the dead locks by defining the problem statement in detail. The intended audience includes all stakeholders, Users, and developers. The detailed low-level design of the simulation of different dead lock scenarios and avoiding the dead locks is provided in this document.

### 1.1 Purpose

The purpose of this document is to describe the low-level design flow of the simulation of different dead lock scenarios and avoiding the dead locks

### 1.2 Document Conventions

TBD (To be continued).

### 1.3 Intended Audience and Reading Suggestions

The document is primarily intended for team members, which consists of trainees under the **Capgemini** Training Program.

### 

**1.4 References**

The references are:

1. System Requirements Specification Document
2. System Specification Requirement

## 2. Detailed System Design

**2.1 Design Descriptions:**

This project design mainly focuses on implementing deadlock concepts. The application will display a main menu to select options such as create a deadlock, prevent deadlock and avoiding the deadlock. The application will display further sub menu options based on selected menu options. The application will validate the menu options at each level. If any incorrect option or entry by the user should display an error.

For simulating deadlock few processes will be created and ran to demo the deadlock scenario. A different set of processes will be created and run to display the demo of preventing deadlock. A set of processes are run to simulate deadlock detection. Implement banker’s algorithm for deadlock detection.

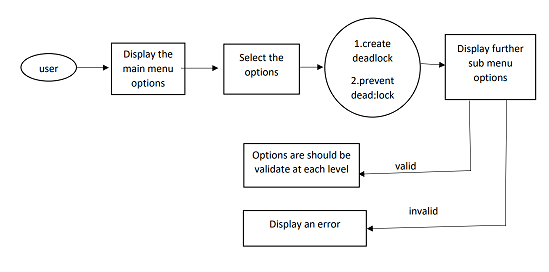
**Main menu:**

User gets options to open the options selection menu or to exit the program.

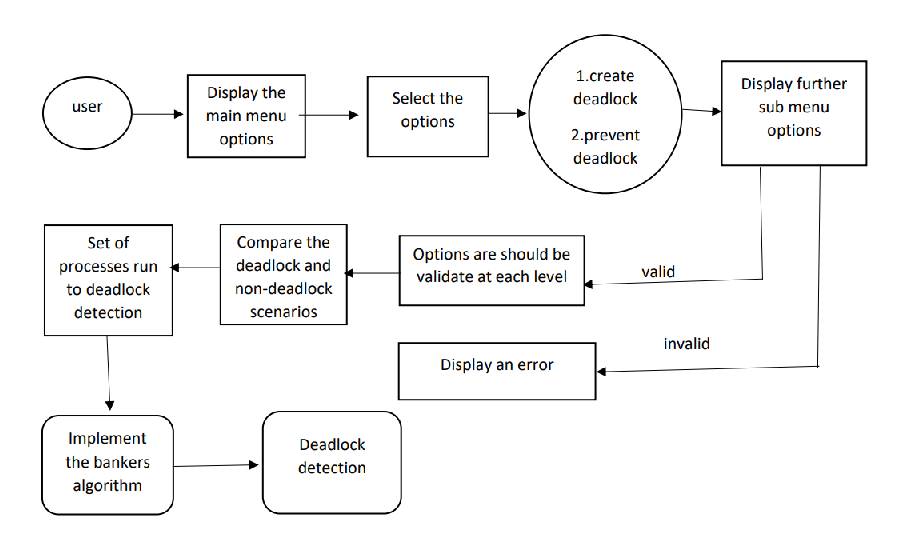
**Selection menu:**

Users get the option to choose the different deadlocks stages (deadlock creation, deadlock prevention) or they can move back to main menu or can exit the program.

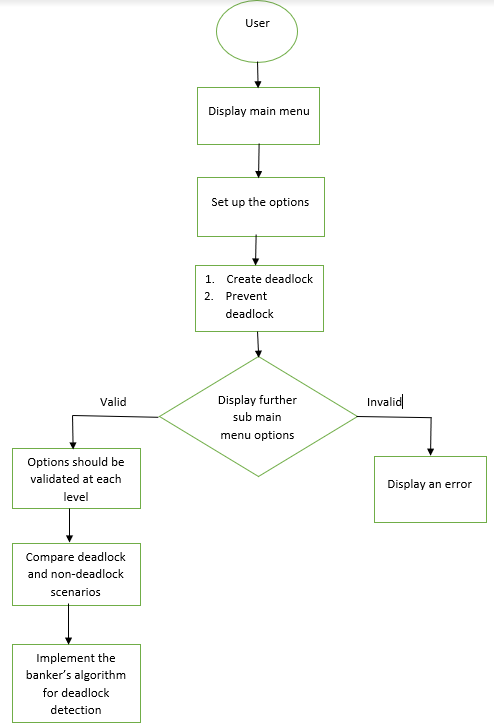
**DFD LEVEL - 0:**



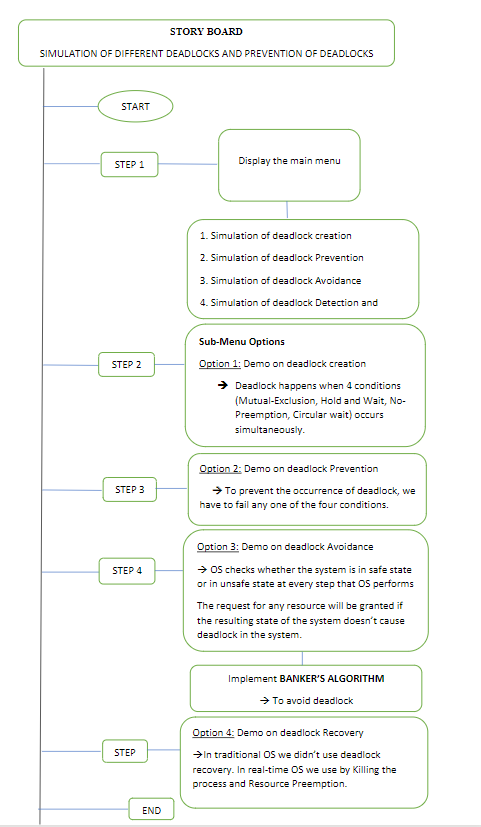
**DFD LEVEL - 1:**



**2.2 Flowchart**



**2.2.3 Storyboard**



**2.3 Modules**

**2.3.1** **main():**

| **Name** | main | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | NA | Initial value | NA |
| **Output** | Return value type | NA | - | - |
| **Description** | The main function will display the user options to the clients | | | |
| **Pseudo Code** | 1. Display the main menu  2. Wait for user options  3. Based on user input it will move to another | | | |

**2.3.2 main\_menu():**

| **Name** | Main\_menu | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | NA | Initial value | - |
| **Output** | Return value type | NA | - | - |
| **Description** | This list of options create deadlock, prevent deadlock, deadlock avoidance, deadlock detection | | | |
| **Pseud**  **Code** | 1. Display the main menu  2.wait for user options | | | |

**2.3.3 Deadlockcreate():**

| **Name** | Deadlockcreate | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | NA | Initial value | - |
| **Output** | Return value type | NA | - | - |
| **Description** | Deadlock happens when 4 conditions (mutual\_exclusion,hold and wait,No\_preemption,circular wait) occurs simultaneously | | | |
| **Pseud**  **Code** |  | | | |

**2.3.4 Deadlockprevent():**

| **Name** | Deadlockprevent | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | Initial value | - | - |
| **Output** | Return value type | NA | - | - |
| **Description** | To prevent the occurance of deadlock, we have to fail any one of the condition | | | |
| **Pseud**  **Code** |  | | | |

**2.3.5 Deadlockavoidance():**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | Deadlockavoidance | | | |
| **Input** | Parameter Name | Intial value | - | - |
| **output** | Return value type | - | - | - |
| **Description** |  | | | |
| **Pseud**  **Code** |  | | | |

**2.5 Use Case Diagram**

**2.6 Design and Implementation Constraints**

The system is built using the C language.

### 2.7 Security

NA