

PROJECT BY : MUHAMMAD RAMEEZ

ROLL NO : 23F-0636

SECTION : BCS-4B

SUBMITTED TO : SIR HASSAN AHMAD

# Operating System Project Documentation

# Overview

This is a custom operating system project that implements various system functionalities and user applications. The project is designed to work on both Windows and Linux platforms, featuring a colorful command-line interface and multiple interactive applications.

System Requirements

- Windows 10 or Linux operating system

- C++ compiler (g++ or compatible)

- Make utility (for building the project)

Project Structure

The project consists of several key components:

Core System Files

- `start.cpp`: Main system interface and task manager

- `time.cpp`: System time management

- `messagebox.cpp`: System message handling

- `stopwatch.cpp`: Stopwatch functionality

- `calendar.cpp`: Calendar application

# Applications

1. \*\*Calculator\*\* (`calculator.cpp`)

   - Basic arithmetic operations

   - Memory management

   - Error handling for division by zero

2. \*\*Tic Tac Toe\*\* (`tictactoe.cpp`)

   - Single player vs Computer

   - AI-based moves

   - Score tracking

   - Colorful interface

3. \*\*Banking System\*\* (`banking.cpp`)

   - Account management

   - Transaction handling

   - Balance tracking

4. \*\*File Management\*\*

   - `createfile.cpp`: File creation utility

   - `deletefile.cpp`: File deletion utility

5. \*\*Games\*\*

   - `hangman.cpp`: Hangman word game

   - `guessgame.cpp`: Number guessing game

6. \*\*Utility Programs\*\*

   - `stringlen.cpp`: String length calculator

   - `length.cpp`: Length conversion utility

   - `findprime.cpp`: Prime number finder

   - `factorial.cpp`: Factorial calculator

   - `binarysearch.cpp`: Binary search implementation

Features

### System Interface

- Colorful command-line interface

- Real-time system time display

- Available RAM monitoring

- Task management system

- Kernel mode access

### Memory Management

- Shared memory implementation

- Process memory tracking

- Memory allocation/deallocation

### User Interface

- ANSI color support for both Windows and Linux

- Clear screen functionality

- Formatted output display

- Interactive menus

# Building and Running

*# For Linux*

make

```

Running the System

```bash

*# Start the main system*

g++ start.cpp -o start -lpthread

./start 2048 8 256

```

# Color Scheme

The system uses the following color codes:

- Red: Error messages

- Green: Success messages and positive outcomes

- Yellow: Prompts and warnings

- Blue: Information display

- Magenta: Computer-related messages

- Cyan: System headers and titles

# Memory Management

- Each application uses shared memory for process tracking

- Memory is allocated and deallocated properly

- System resources are managed efficiently

Error Handling

- Input validation

- Division by zero protection

- File operation error handling

- Memory allocation checks

Cross-Platform Compatibility

- Conditional compilation for Windows/Linux

- Platform-specific color codes

- System command compatibility

Security Features

- Input validation

- Memory protection

- Process isolation

- Resource management

Future Improvements

1. Add more applications

2. Implement file system

3. Add user authentication

4. Enhance memory management

5. Add networking capabilities

Contributing

Feel free to contribute to this project by:

1. Forking the repository

2. Creating a feature branch

3. Making your changes

4. Submitting a pull request

License

This project is open source and available under the MIT License.

Authors

- MUHAMMAD RAMEEZ

Acknowledgments

- Inspired by various operating system concepts

- Built with C++ standard library

# SCREENSHOTS



















 









