

## OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

Introduction

Software Requirement

Specifications

Requirements
Non-Functional

Requirements
Use-Case Diagram

Technolog Stack

OOA

Inherited Classes
Class Diagram

Conclusion

Dependencies Future Aspects

# OOPs Project Presentation

Snakes and Ladders

Y. Chauhan<sup>1</sup> S. Ladhe<sup>1</sup> S. Chinchkar<sup>1</sup> S. Kumar<sup>1</sup>

<sup>1</sup>Computer Science and Engineering Indian Institute of Information Technology -Vadodara, International Campus Diu

18th October, 2023



# Table of Contents

OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

Introduction

Requirement Specifications

Functional

Non-Functiona Requirements

Use-Case Diagram

Technology

OOA

Identifying Classe Inherited Classes Class Diagram

Conclus

Dependencies Future Aspects Conclusion 1 Introduction

2 Software Requirement Specifications

Functional Requirements

Non-Functional Requirements

Use-Case Diagram

3 Technology Stack

4 00A

Identifying Classes

Inherited Classes

Class Diagram

5 Conclusion

Dependencies

■ Future Aspects

Conclusion



# Introduction

## OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introduction

Software Requirement Specifications Functional Requirements

Non-Functional Requirements Use-Case Diagram

Technology

## $\bigcirc\bigcirc$

Identifying Classes Inherited Classes Class Diagram

## Conclusion

Dependencies Future Aspects Conclusion

# Snakes and Ladders

- The Snakes and Ladders Game is a digital recreation of the classic board game. The primary aim is to provide an enjoyable and interactive gaming experience for players of all ages.
- Snakes and Ladders is one of the most recognizable board games today. Originated in ancient India around the 13th century AD, the game was designed to teach children the cause and effect of good and bad deeds.



# Objectives

## **OOPs Project** Presentation

Yuvraj, Shrevas. Sneha, Surai

# Introduction

- Our project aims to demonstrate the effectiveness of an Object-Oriented approach in solving complex problems.
- We'll showcase how abstraction and inheritance enhance efficient product design.
- Object-oriented concepts streamline debugging and optimize the CI/CD Pipeline.
- The Web-App interface ensures compatibility across all devices and eliminates support concerns.
- A simple Web-App guarantees playability on any device with internet access and a browser.



# SRS and Use Case Diagram

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introduction

## Software Requirement Specifications

Specification

Non-Functiona

Use-Case Diagran

Ose-Case Diagran

## $\bigcirc\bigcirc$

Identifying Classes Inherited Classes

### Conclusion

Dependencies Future Aspects

- 1. Functional Requirements
- 2. Non-Functional Requirements
- 3. Use Case Diagram



# Functional Requirements

## OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introductio

Requirement Specifications

Functional Requirements

Requirements

- Use-Case Diagrai

OOA

Identifying Classes Inherited Classes Class Diagram

# Conclusion

Dependencies Future Aspects Conclusion ■ **Die Rolling:** Implement random die roll functionality (1-6).

- **Player Movement:** Move the player's game piece based on the die roll.
- Consecutive 6s Rule: Detect three consecutive 6s and void the last 6.
- **Normal Block:** Move the player to the designated block.
- Snake Head Block: Move the player to the corresponding snake's tail block.
- Ladder Bottom Block: Move the player to the corresponding ladder's top block.



# Non-Functional Requirements

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introductio

Requirement
Specifications
Functional

### Non-Functional Requirements

Use-Case Diagra

# Stack

Identifying Classes

## Conclusion

Dependencies Future Aspect  User Interface: Intuitive and visually appealing user interface.

- Performance: Smooth game play with responsive controls.
- **Compatibility:** The game should run on popular web browsers.
- **Security:** Ensure data privacy and prevent cheating.



# Use Case Diagram

OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

Introduction

Software Requirement

Specifications

Requirements

Non-Function Requirements

Use-Case Diagram

Technology

 $\bigcirc\bigcirc$ 

Identifying Classes
Inherited Classes
Class Diagram

Conclusion

Dependencies

Future Aspect Conclusion

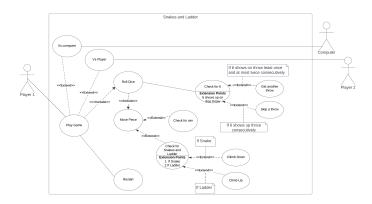


Figure: Use Case Diagram



# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introduction

Requirement

Specification

Requirements

Requirements

Use-Case Diagram

Technology

# Stack

Identifying Classes

Inherited Classes Class Diagram

### Conclusion

Dependencies
Future Aspect

# 1. Front End

# Front End

- React JS
- CSS



# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introduction

Software Requirement

Specifications

Requirements
Non-Functiona

Non-Functiona Requirements

Use-Case Diagrai

# Technology Stack

OOA

Inherited Classes

### Conclusion

Dependencies
Future Aspect

# 1. Front End

# Contributors

- Shreyas Ladhe
- Sneha Chinchkar



# **OOPs Project** Presentation

Yuvraj, Shreyas, Sneha, Suraj

# Technology Stack

1. Front End

2. Back-End

# Back End

- Django
- Python



# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introduction

milioduction

Requirement

Specification

Formation of

Requirements

Non-Function Requirements

Use-Case Diagram

Technology

# Stack

Identifying Class Inherited Classe

Inherited Classes Class Diagram

### Conclusion

Future Aspect Conclusion

- 1. Front End
- 2. Back-End

# Contributors

- Yuvraj Chauhan
- Suraj Kumar



# **OOPs Project** Presentation

Yuvraj, Shreyas, Sneha, Suraj

Technology

# Stack

- 1. Front End
- 2. Back-End
- 3. Version Control

# Version Control

- Git
- GitHub



# **OOPs Project** Presentation

Yuvraj, Shreyas, Sneha, Suraj

Technology

# Stack

# 1. Front End

- 2. Back-End
- 3. Version Control

# Contributors

- Yuvraj Chauhan
- Shreyas Ladhe
- Sneha Chinchkar
- Suraj Kumar



# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introduction

6.6

Requirement

Specification

Requirements

Non-Functiona Requirements

Requirements

Technology

# Stack

Identifying Classes
Inherited Classes
Class Diagram

### Conclusion

Dependencies
Future Aspects
Conclusion

- 1. Front End
- 2. Back-End
- 3. Version Control
- 4. UI/UX

# UI/UX

Canva



# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introductio

Software

Specification

- . . .

Requirements

Non-Functiona Requirements

Kequirements

Technology Stack

004

Identifying Classes
Inherited Classes
Class Diagram

### Conclusion

Dependencies
Future Aspect

- 1. Front End
- 2. Back-End
- 3. Version Control
- 4. UI/UX

# Contributors

Shreyas Ladhe



# Object Oriented Analysis

## OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introduction

- Requirement Specifications
- Specification
- Requirements Non-Functional
- Requirements
- Ose-Case Diagra

# OOA

Inherited Classes

### Conclusion

Dependencies

Future Aspect Conclusion

- 1. Identifying Classes
- 2. Inherited Classes
- 3. Class Diagram



# **Identifying Classes**

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introduction

- Software
- Requirement
- Specification
- Functional
- Non-Functional
- ....
- Ose-Case Diagran
- -----

# Identifying Classes

Inherited Classes

### Conclusio

Dependencies Future Aspects

- Player
- Dice
- Cell
- Board
- Game



# Inherited Classes

## OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

## Introductio

Software

Specification

5pecification

Requirements

Requirements

Use-Case Diagran

004

Identifying Classes

Inherited Classes

Conclusio

Conclusion

Future Aspect Conclusion

- Inherited Class from Cell:
  - firstCell
  - Jumper
  - lastCell
- Inherited Class from Jumper
  - Snakes
  - Ladders



# Class Diagram

## **OOPs Project** Presentation

Yuvraj, Shreyas, Sneha, Suraj

Class Diagram

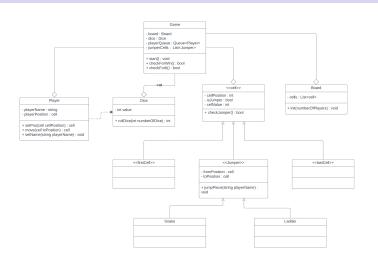


Figure: Class Diagram



# Dependencies

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introductio

- Requirement
- Specification
- Requirements Non-Functional
- Requirements
  Use-Case Diagram
- Use-Case Diagra

# 004

Inherited Classes

### Conclusion

# Dependencies

Future Aspec Conclusion

- Requires a modern browser to run.
- Multiplayer functionality is not available
- Interface might get overloaded due to multiple requests (dice roll).



# **Future Aspects**

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introductio

- Requirement
- Specifications
- Requirements Non-Functional
- Requirements
- Ose-Case Diagrai

# OOA

Inherited Classes

# Conclusio

Dependencies
Future Aspects

Multiplayer functionality

- Save game progress and loading saved game
- Incorporating Cloud hosting and saving game progress on Cloud.



# Conclusion

# OOPs Project Presentation

Yuvraj, Shreyas, Sneha, Suraj

### Introductio

Software Requirement Specifications

Functional Requirements Non-Functional

Non-Functional Requirements

Technology

# OOA

Identifying Classes
Inherited Classes
Class Diagram

### Conclusion

Dependencies Future Aspects Conclusion

# Conclusion

Our Object-Oriented "Snakes and Ladders" project demonstrates the power of OOP principles for efficient, maintainable games. Future plans include multiplayer, saved game progress, and cloud integration.