Team 3 (Pong Game)

1. **Leo Pan**
   * **Student ID**: 030025552
   * **Email**: Leo.Pan02@student.csulb.edu
2. **Darren Ammara**
   * **Student ID**: 029035043
   * **Email**: Darren.Ammara01@student.csulb.edu
3. **Nathalia Coronado**
   * **Student ID**: 030127924
   * **Email**: nathalia.coronado01@student.csulb.edu
4. **Evan Gonzalez**
   * **Student ID**: 030209229
   * **Email**: evan.gonzalez01l@student.csulb.edu
5. **Jaime Lopez**
   * **Student ID**: 030564961
   * **Email**: jaime.lopez01@csulb.edu
6. **Martin Silva**
   * **Student ID**: 030854159
   * **Email:** martin.silva01@student.csulb.edu

## Preface

| Version | Date | Changes |
| --- | --- | --- |
| 1.0 | 11/17/2024 | Initial Version |
| 1.1 | 11/18/2024 | Introduction and Requirements |
| 2.0 | 11/19/2024 | Completed Case Flows and User Stories |

### Purpose

This document serves as a comprehensive guide for the development and comprehension of the software project entitled “Pong.”

### Audience

This document is intended for the following stakeholders: project managers, developers, testers, and all other individuals involved in the project lifecycle.

## Introduction

### Project Overview

“Pong” is a simple local computer game meant to be played by oneself against a computer AI opponent, or with another human player on the same computer keyboard.

### Project Goals

* Generate Entertainment
* Allow individual practice of the game mechanics
* Encourage competitiveness around the game mechanics

## Glossary

* **PyGame**: Game engine for Python.
* **Sprite**: One individual game class or object

## User Requirements and Use Cases

### User Stories

(A collection of user stories that apply to the project.)

1. As a casual gamer, I want to play against my friend locally on the same computer so that we can enjoy some friendly competition and have fun.
2. As a beginner player, I want to play against a bot on easy mode so that I can practice and get better without getting frustrated.
3. As a busy student, I want the game to start quickly without too many options so that I can play a match during a short break.
4. As someone who likes a challenge, I would like to try to beat a Bot on the hardest difficulty there is to prove my skill.
5. As someone who wants to get people into retro games, I would like a game to play with my friends that reminds me of the original pong experience
6. As a competitive person, I would like an easily accessible game to compete with my friends in our free time.
7. As a nostalgic person, I would like an authentic version of the game pong that reminds me of my childhood
8. As a full time worker I would like to enjoy a fun game to take the stress off from work.
9. As an older gentleman, I would like to play a game that doesn’t require me to memorize various control options.
10. As a competitive person I would like to show off proof of my skills on the leaderboard to my peers.

### Use Case: Adding a new score to the Leaderboard

| Identifier | UC-1 Add new EASY leaderboard score |
| --- | --- |
| Purpose | Update the local leaderboard to show the user’s score against the EASY AI. |
| Requirements | User Story #10 |
| Development Risks | None |
| Pre-conditions | Player has initiated a single-player session against the EASY AI. |
| Post-conditions | New record is added to the leaderboard. |

***Table 1: Typical Course of Action***

| Seq# | Actor’s Action | System’s Response |
| --- | --- | --- |
| 1 | Scores points against the AI |  |
| 2 | Loses 3 points to the AI |  |
| 3 |  | Renders Game over Screen and exit buttons. |
| 4 |  | Checks if the player’s score is higher than any previous scores |
| 5 |  | Check passes and leaderboard is updated. |
|  |  |  |

***Table 2: Alternate Course of Action***

| Seq# | Actor’s Action | System’s Response |
| --- | --- | --- |
| 1 | Scores points against the AI |  |
| 2 | Loses 3 points to the AI |  |
| 3 |  | Renders Game over Screen and exit buttons. |
| 4 |  | Checks if the player’s score is higher than any previous scores |
| 5 |  | User has not scored enough points and the leaderboard is left un-updated. |
|  |  |  |

***Table 3: Exceptional Course of Action***

| Seq# | Actor’s Action | System’s Response |
| --- | --- | --- |
| 1 | Scores points against the AI |  |
| 2 | User exits game before losing 3 points to the AI |  |
| 3 |  | Returns to the Main Menu without ever checking the Leaderboard and the User’s score is lost. |
|  |  |  |
|  |  |  |
|  |  |  |

## System Architecture

### Components

1. **Game Engine:** PyGame
2. **Backend**: Text files for leaderboard database
3. **Language:** Python

### Architectural Pattern

We used the Model-View-Controller architectural design for our project. The following is a model for it.

