# DFD specification Document

# Become Pac-Man

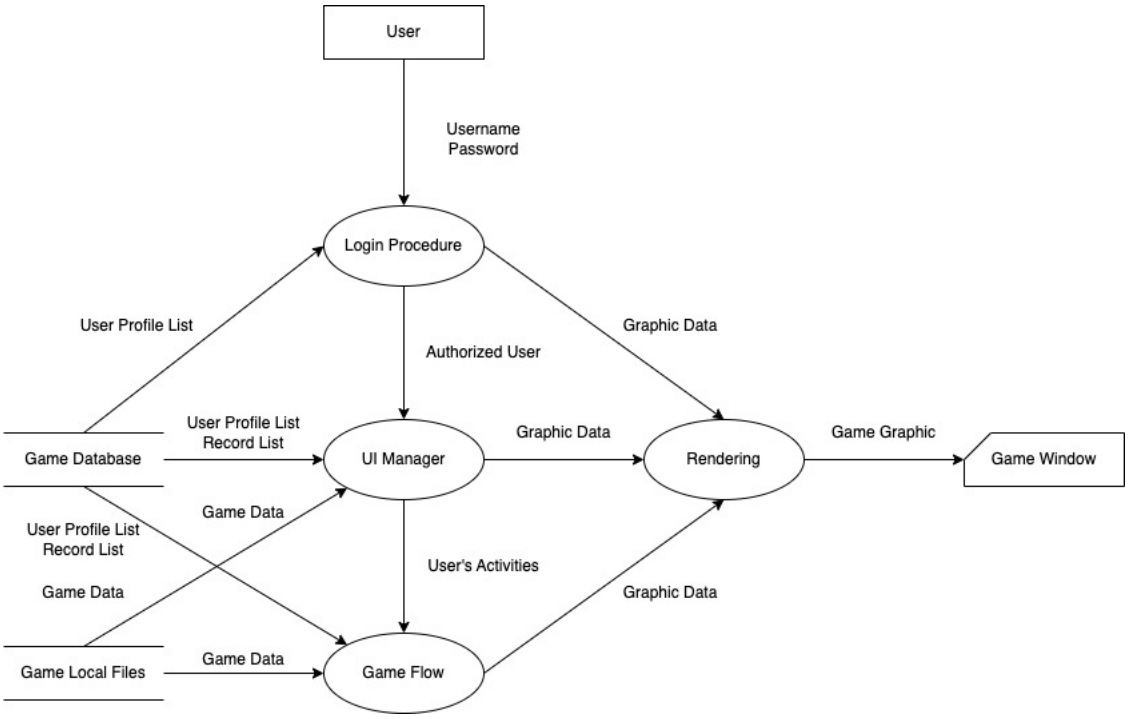
# version: 1

Group F8
1155127434 HO Chun Lung Terrance
Department of Philosophy, The Chinese University of Hong Kong
1155143519 WOO Pok
Department of Physics, The Chinese University of Hong Kong
1155157839 NG Yu Chun Thomas
Department of Computer Science and Engineering, The Chinese University of Hong Kong
1155157719 LEUNG Kit Lun Jay
Department of Computer Science and Engineering, The Chinese University of Hong Kong
1155143569 MOK Owen
Department of Mathematics, The Chinese University of Hong Kong

February 27, 2023

## Contents

# 1 High-Level Context Diagram




Our game data has the flow as above:

At first, the users need to input their username and password into the login procedure, and the input data will be matched with the Game Database to authorize the user. After that, the authorized user will be directed to the UI Manager. The details will be described at UI DFD later. The UI Manager receives user's input and data from Game Database and Game Local Files, and output user's activities to Game Flow to run the game. The details of Game Flow will be described later as well. The functions mentioned above will all output graphic data to Rendering in the Unity engine and output as visible graphic to the Game Window for users to view.

# 2 Feature Diagrams

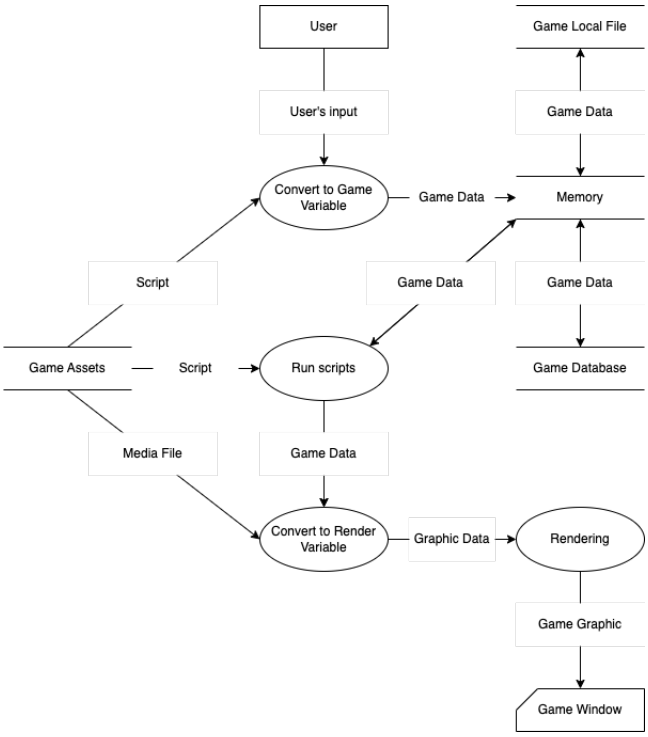
## 2.1 Game Rendering

### 2.1.1 Description

Game render as follows. The user's input will be converted to game variables by the scripts stored in the Game Assets, and stored to the local memory. The memory has data exchange between the Game Local Files and the Game Database. While the scripts in the Game Assets are running, they receive and also write game data to the memory, which can also be sednt to the Game Local Files and the Game Database. Running scripts will send game data to the Unity engine, and media

files in the Game Assets like images, videos or animations, will be loaded to the engine too. These variables will be converted to render variables and output as graphic data to the renderer to do rendering. Lastly, the actual graphic will be shown on the Game Window to the user.

### 2.1.2 DFD




## 2.2 Game Flow

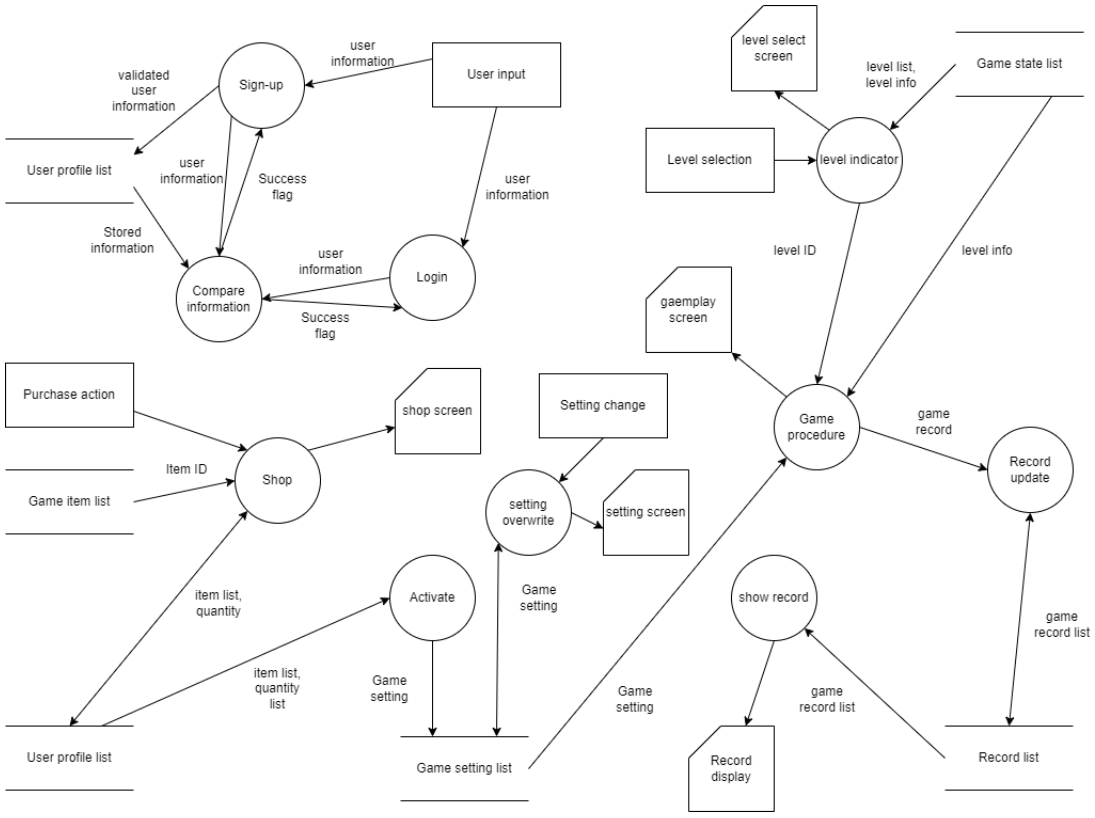
### 2.2.1 Description

**Login/Sign-up:** The Sign-up function and Login function both gets user information from user input, then send the information to the compare function. The compare function will retrieve stored user information from user profile list for validation of inputted information. Once success (or fail), it will send back a success flag to indicate the result of validation to sign-up function or Login function, depends on the current procedure. For Sign-up function, if the validation succeed, it will push the inputted information to user profile list as a new storage. For Login function, it will progress to the title page frame.

**Title screen page:** There will be several functions under Title screen page, including Shop function, Activate function, Setting overwrite function, level selection function, Record update function, and show record function. For Shop function, it will retrive item ID from Game item list, the item list from user profile list, and provide information to shop display. After purchase action, the item purchased and the respective quantity will be stored to user profile list. Then, the activate

function will be called to get the updated item list from user profile list, manipulate the data into specific game setting, and put them into game setting list. When setting change is performing, the setting overwrite function will be called and game setting will be retrieved from game setting list. The game settings will be send to setting screen, and once game setting changes are completed, the updated game setting will be stored into game setting list. During level selection, the level indicator function will be called, then level list and level information will be retrieved from game state list. The data will be sent to level select screen for displaying, and once level selection is completed, selected level ID will be sent to game procedure. The game procedure will retrieve level information according to the recieved level ID, with game setting stored in game setting list, to perform the game, sending gameplay information to the gameplay screen. After the game procedure, game record will be generated and be sent to the record update function immediately, called right after the game ends. The record update function will simultaneously retrieve game record list from record list to update the list of game record. It will then store the updated record to record list. Once the show record function is called, the stored game record list will be retrieved from record list, and be sent to record display.

### 2.2.2 DFD

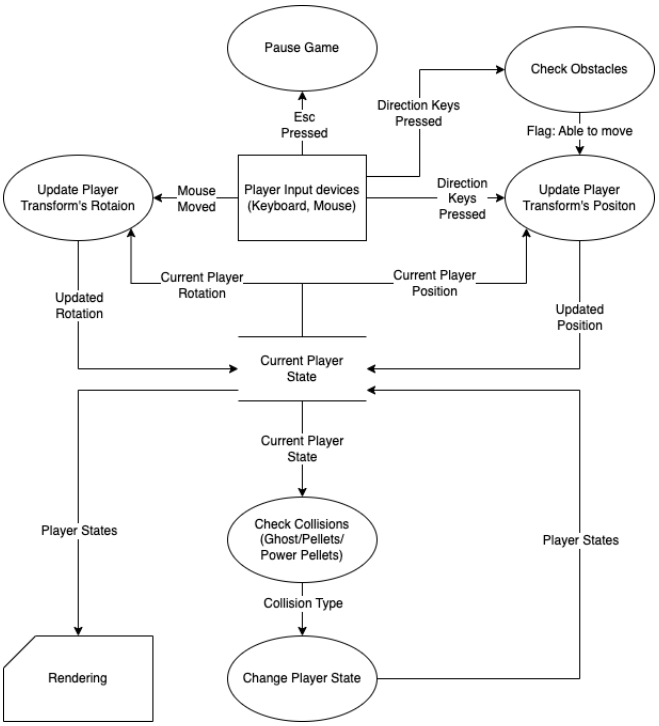


## 2.3 Player Control

### 2.3.1 Description

For player control, the game will receive signals from the user's input device and perform different actions. For example, the user can move Pacman and change its vision using a keyboard and mouse, also one can pause the game by pressing the "Esc" key. On moving, the game checks for any collisions to update the player's state, such as collisions with ghosts will decrease lives, collisions with power pellets will enter the powered state, and collisions with walls refuse the player to move along the direction. The player state (transform, "isPowered", remaining pellets, etc.) will be saved and output to the engine.

### 2.3.2 DFD



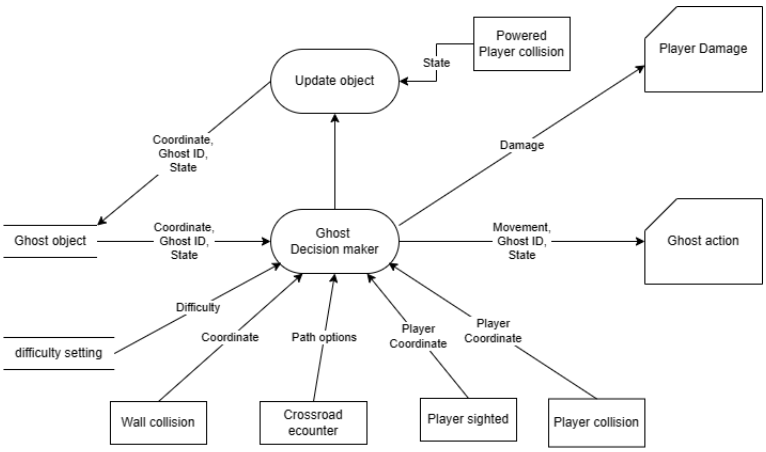

## 2.4 Enemy AI

### 2.4.1 Description

During the game, for each ghost object, it will perform all of its action based on a central function "Ghost decision maker". This function will take data from two storages: First, it will take the coordinates, Ghost ID, and ghost state (alive or dead, or other states) from the data store "Ghost object", which should be created for each ghost by each match; Secondly, it will take the game difficulty from "difficulty setting" to consider ghost action complexity. With the above data, the

Ghost Decision Maker will anticipate any input events received by the ghost object, including: Wall collision, Crossroad encounter, Player sighted, and Player collision. From each of the five event inputs, the Decision Maker will output its action based on the data from Ghost object and Difficulty Setting, and output the movement, Ghost ID, and Ghost state of the ghost object into a ghost action. It can also ouput a damage value as the Player Damage (such as when colliding with a non-powered player). For each output of the Decision maker, it will also output an Update that will update the data store of Ghost Object. As for when encoutering a Powered player Collision event, this will directly update the Ghost state of the object from "alive" into "dead" as a way to kill the ghost, the Decision maker will then output Ghost actions based on this new state (such as stopping and playing death animation.)

### 2.4.2 DFD




## 2.5 Game UI

### 2.5.1 Description

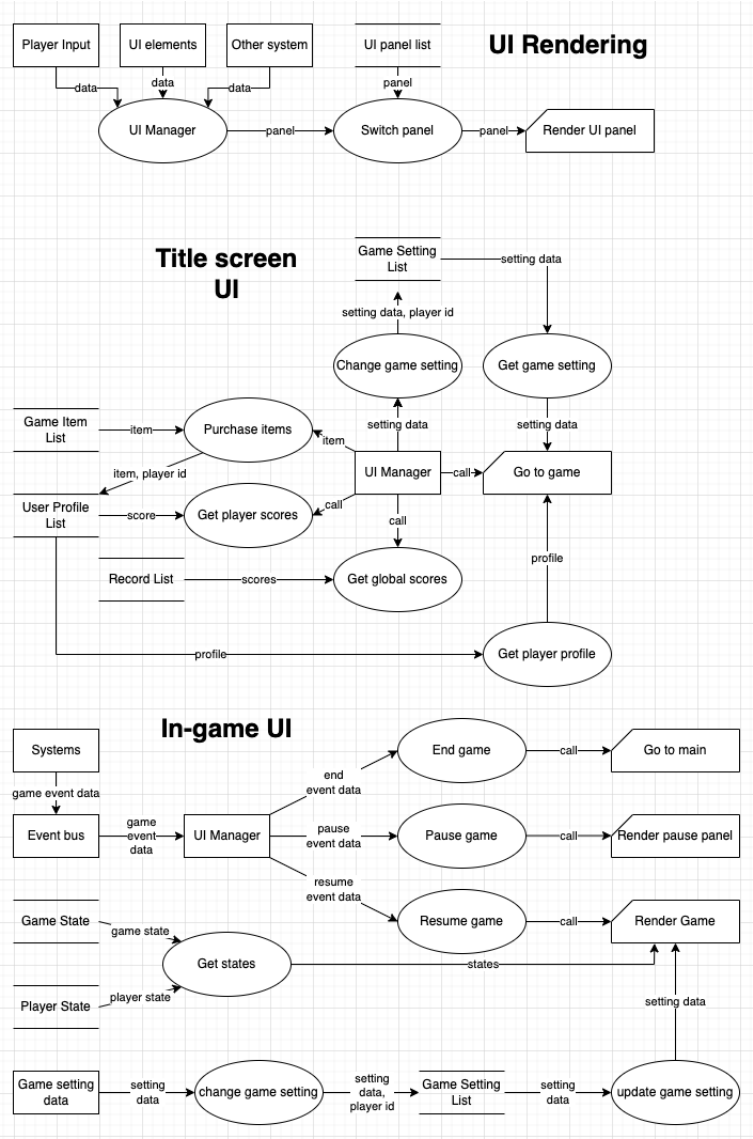
**UI Rendering:** The UI Manager handles input and output of the UI. For example, during game, a 'game end' event can be invoked when a player choose to quit game after pausing or a ghost killing the pacman. For the first case, a button UI element calls UI Manager to end a game; for the second case, an other system calls UI Manager. After that, the UI Manager will choose to show and render the UI panel by calling the SwitchPanel() function. UI panel list stores a list of displyable panels where a panel contains a list of UI elements. The panel list may contain panels such as title main screen panel, shop panel and game pause panel. A UI panel may contains elements such as button, text and slider.

**Title Screen UI:** As mentioned before, UI Manager handles input and rendering of UI. The UI Manager may call different functions according to the inputs. Change game setting function change the player game setting data such as music volume and graphic setting. It will update the

Game Setting List which stores all player setting on the database. Our Pacman has a shop system where player can buy skin or other game items. The game item list stores all game item while user profile list stores all user profile. A user profile contains the player's owned game items and score. A record list stores all players highest score. the world record will be displayed on the title screen for players to compare. Everytime when the game is started, it will fetch the player game setting and the player profile for the game rendering.

**In-game UI:** During game, a game event, such as game pause and game end, can be called by different systems, such as UI Manager and Player Controller. For instance, a game end event can be invoked by the UI Manager when a player clicked the end button in pause panel or by the Player Controller when a player health is below 0. UI Manager handles all the game events in game and react according to it. When game is running normally, there are also data needed to be rendered, such as player health and scores. Those data will be fetched from Game State and Player State and keep updating. If a player would like to change setting in game after pausing, the new game setting data will be passed to the change game setting function and it would update the Game Setting List Database. When the game resumes, it will first update the game setting and then continue the game.

### 2.5.2 DFD


UI Rendering
Player Input
UI elements
Other system
UI panel list
data
data
data
panel
UI Manager
panel
Switch panel
panel
Render UI panel
Title screen UI
Game Setting List
setting data
setting data, player id
Change game setting
Get game setting
Game Item List
item
Purchase items
item
setting data
setting data
item, player id
UI Manager
call
Go to game
User Profile List
score
Get player scores
call
call
Record List
scores
Get global scores
profile
profile
Get player profile
In-game UI
Systems
End game
call
Go to main
game event data
end event data
Event bus
game event data
UI Manager
pause event data
Pause game
call
Render pause panel
resume event data
Game State
game state
Resume game
call
Render Game
Get states
states
Player State
player state
setting data
Game setting data
setting data
change game setting
setting data, player id
Game Setting List
setting data
update game setting