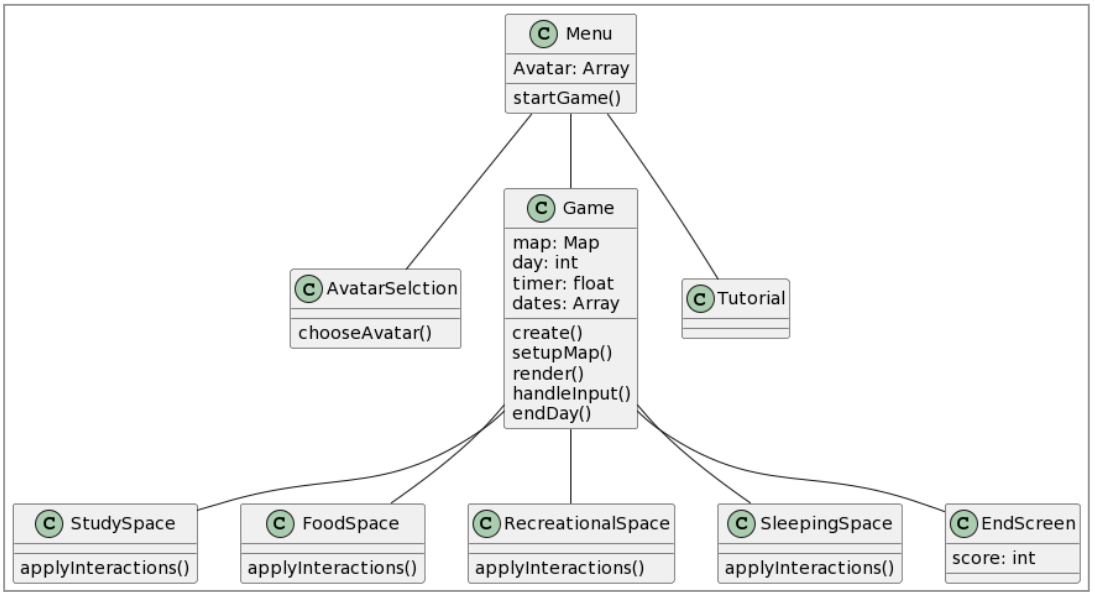
Structural Diagram

I. Class Diagram: Screen

Using PlantText, two UML class diagrams were created to help us visualise the different elements of the game and the relationships between them. The first class diagram represents the screens of the game and includes the main methods and attributes relating to the more structural, background aspects of the game. The player will begin at the menu with an option to move to a tutorial screen to learn the objectives and controls of the game, to an avatar selection screen to select their character design, or to begin playing. The Game screen is the main screen of the game where the map will be visible and the core gameplay loop will take place. Each of the activities the player can complete - Studying, eating, recreational activities & sleeping - will be all accessible from the main game screen and the applyInteractions() method is responsible for outputting to screen the text relating to each activity. FInally, once the player has completed 7 days, the EndScreen will be shown with the player's statistics & final score.

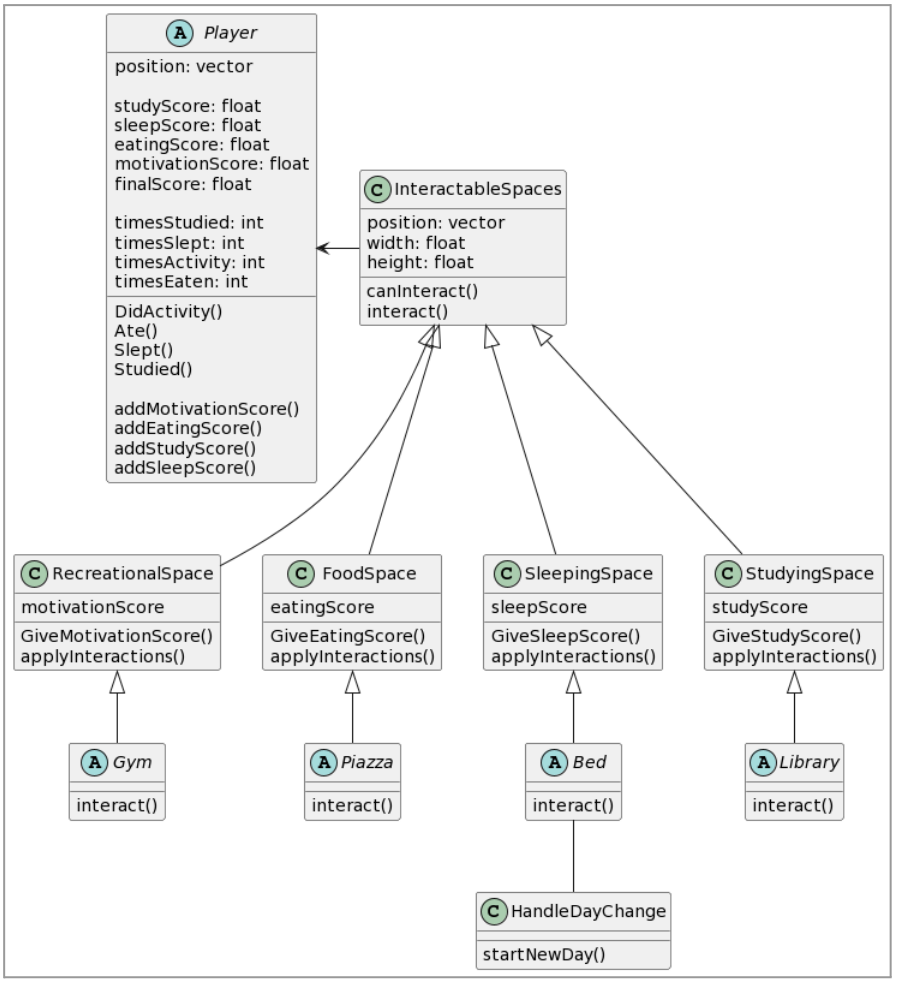


Linking to the requirements elicited previously:

The tutorial screen, accessible from the main menu satisfies the requirements “UR_IN_GAME_HELP” and “NFR_PLAYER_GUIDANCE” by providing the player with help and instructions to play the game.
The Avatar screen corresponds to the requirement “UR_CUSTOMISATION” by allowing the player to change the appearance of their in-game avatar.
The main game screen will fulfil the “FR_MAP” requirement by providing the playable area for the character to move around in, and with correct implementation, shall also satisfy the requirements “UR_MOVEMENT” and “UR_RESTRICTED_MOVEMENT” by controlling how and where the player moves around.
The activity locations will also correspond to a range of previously determined requirements. Firstly, “FR_MULTIPLE_LOCATIONS” outlines the need to have separate locations for eating, sleeping, studying and recreational activities, which are all governed by the respective ‘xSpace’ classes. These classes allow multiple instances of each activity to be made, giving the user some choices in how they wish to complete certain tasks. For example, the player has the option to interact with various recreational activities such as going to the gym, going into town etc. This will also help satisfy the requirements

“NFR_ENGAGEMENT” and “UR_REPLAY_VALUE” as it allows the user to play the game in several different ways.
In addition, the StudySpace class shall also satisfy the “FR_STUDY_CHOICES” requirement as it shall provide the user some freedom in how long they wish to study for. This in turn allows for the player to develop their own style of playing the game, which may also contribute towards the replayability requirements mentioned above.
Finally the EndScreen shall satisfy the “UR_END_GAME” requirement by providing the player with their score and exam result after 7 days of gameplay.

II. Class Diagram: Objects and User Interaction



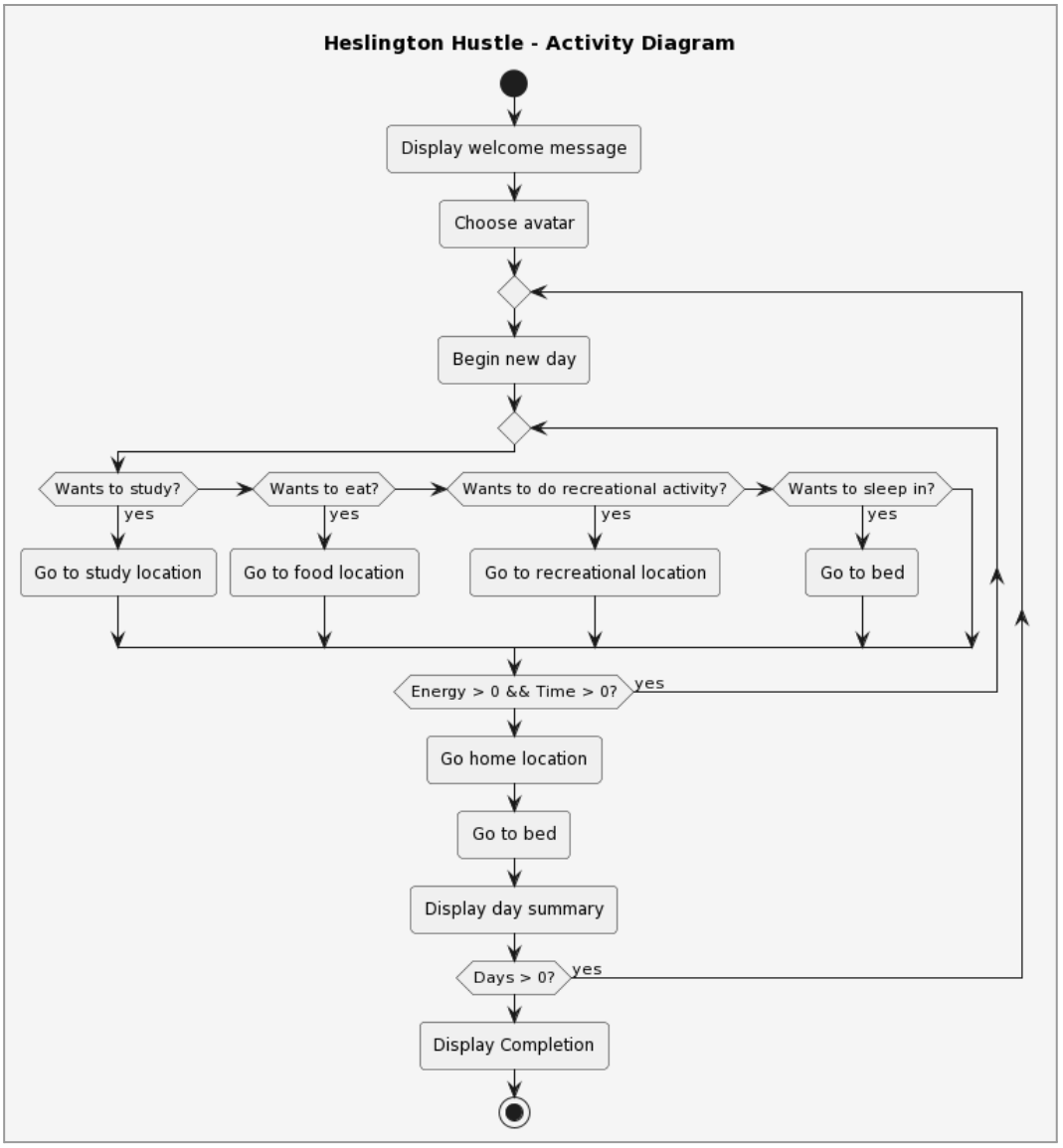
The second UML diagram focuses on the objects the user interacts with, specifically the 4 required activities that the player must complete during the day. These are all instances of their respective “XSpace” class allowing for the creation of multiple spaces to complete each activity. In addition the UML diagram outlines how the scores for each activity will be incremented with the use of the “GiveXScore” method, depending on the activity completed.

For the requirements, the “FR_DAY_NIGHT_CYCLE” is met through both the sleep activity handling the changing of days, and also the applyInteractions() as this method prevents the player from sleeping before a certain hour of the day. The “FR_TRACK_STATISTICS” is also satisfied through the incrementing of each score after an activity is completed. The support for multiple locations for each activity will also help fulfil the replayability-based requirements such as “NFR_ENGAGEMENT” and “UR_REPLAY_VALUE”

Behavioural Diagrams

It was useful to think about how the user would interact with our game in addition to the class diagrams. As a result, we produced an activity diagram to outline the flow of actions that will be performed by the user and a state diagram to illustrate how the game operates and screen changes throughout gameplay. Both diagrams were created using PlantText.

I. Activity Diagram



The activity diagram depicts the flow of actions within the game. The game starts with displaying a welcome message and allowing the player to choose their avatar. Then, the player begins a new day, where they have several options: they can choose to study, eat, engage in a recreational activity, or sleep in. The player continues to perform activities throughout the day, given that their energy and time resources are not completely depleted. Once the day ends, the player returns home, goes to bed, and the day summary is displayed. This process repeats until the player has completed all the days in the game. Finally, upon completing all the days, the game displays a completion message.

The interim version of the diagram starts directly with the user performing actions. This was not ideal as the development went on and ideas were thrown around which ended up with

the creation of a welcome screen as well as the avatar selection feature. Moreover, in between the “Begin new day” and the resources conditional check, there was only a node which says “Perform actions”. For clarification purposes, this was changed to more specific actions that the users would do on the game as seen in the diagram above.

Tracing back to the requirements of the system and how they relate to the activity diagram:

1. Display welcome message and choose avatar: These steps align with the system requirement "NFR_UX," which states that the game shall provide a pleasant user interface. It ensures that the game starts with a welcoming message and allows the player to choose their avatar, enhancing the overall user experience.

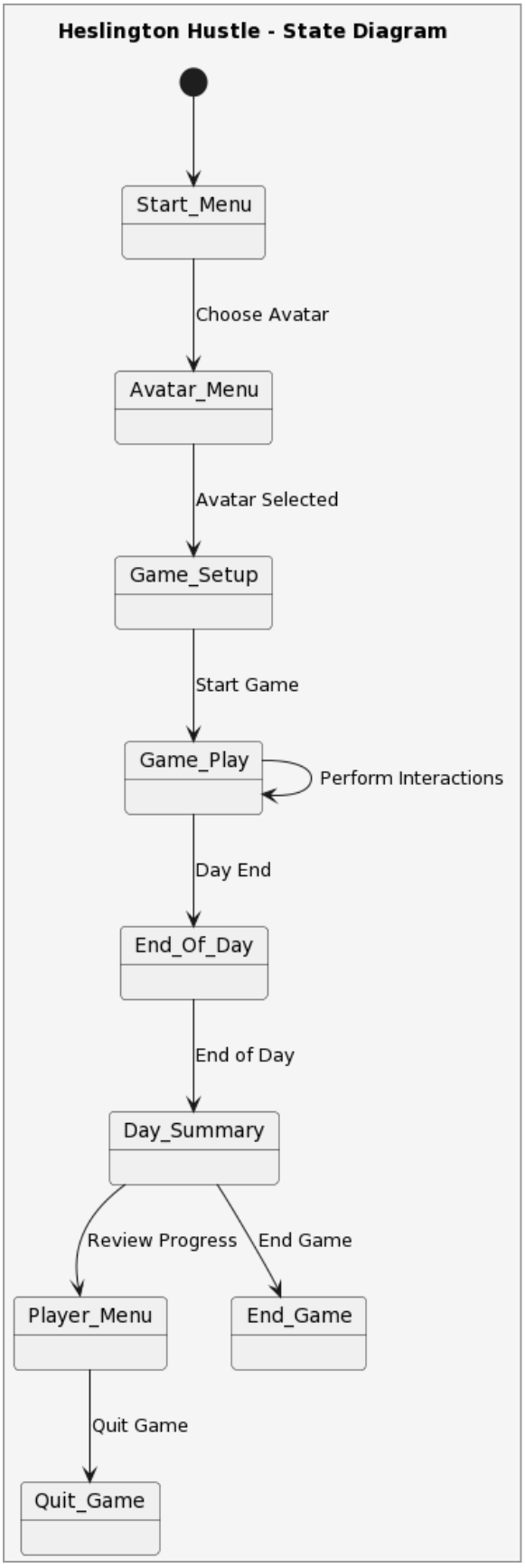
2. Begin new day: This step corresponds to the functional requirement "FR_MAP" and functional requirement "FR_MULTIPLE_LOCATIONS", which states that the game should have a map the user can move around. Starting a new day signifies the progression of time within the game, allowing the player to interact with various locations on the map.

3. Perform activities (study, eat, recreational activities, sleep): These steps fulfill the user requirement "UR_INTERACTION", which states that the user should be able to interact and perform activities at different locations on the map. Each activity option aligns with the user's need to manage their character's daily routine effectively.

4. Display day summary: This step aligns with the user requirement "UR_CLEAR_FEEDBACK" and functional requirement "FR_TRACK_STATISTICS" which states that the user should be able to keep tabs on their game progression. Displaying the day summary allows the player to review their actions and progress within the game.

5. Energy and time management: The decision points in the activity diagram, such as choosing whether to study, eat, engage in recreational activities, or sleep, directly relate to the user requirement "UR_RESTRICTED_INTERACTION." This requirement specifies that the user should not be able to perform activities if they have low energy or time resources. Therefore, the architecture ensures that the player's actions are constrained by their energy and time availability, enhancing the realism and challenge of the game.

II. State Diagram



The state diagram depicts the flow of interactions between a player and a game. The initial state is the "Start Menu," where the player begins the game. From there, the player moves to the "Avatar Menu" to choose their avatar. Once an avatar is selected, the game transitions to the "Game Setup" state, where initial game settings are configured. After setup is complete, the game enters the "Game Play" state, allowing the player to interact with the game world. During gameplay, the player may transition to the "End of Day" state when a day in the game ends. From the "End of Day" state, the game moves to the "Day Summary" state to provide the player with a summary of their progress. The player can then choose to access the "Player Menu" to review settings or quit the game, leading to the "Quit Game" state. Alternatively, the game may transition directly to the "End Game" state if certain end-game conditions are met.

As for the interim version of the diagram, it was initially thought that variables should be included in each state which resulted in the diagram being too convoluted. An overhaul was needed so that unnecessary complexity can be omitted from the diagram. This derived a simplified and clear view of each screen/state as shown in the diagram.

Tracing back to the requirements of the system and how they relate to the state diagram:

1. Start Menu, Avatar Menu, Game Setup, Game Play, End of Day, Day Summary, Player Menu, Quit Game, End Game states: Each of these states corresponds to different stages or actions within the game, fulfilling the non-functional system requirement for providing a structured and intuitive user interface ("NFR_UX"). The architecture ensures that the player can navigate through different menus, perform actions within the game, review progress, and manage game settings, aligning with the user's expectation for a cohesive and user-friendly experience.

2. Avatar Selection: The state diagram includes a specific state for choosing the player's avatar, which aligns with the user requirement for allowing the player to select

their avatar ("UR_CUSTOMIZATION"). This requirement ensures that the player has agency and customization options within the game, contributing to a more personalised gaming experience.

3. Game Setup and Game Play states: These states correspond to the functional requirement for providing a map the user can move around ("FR_MAP"). The game setup stage allows the player to prepare for gameplay, while the game play stage enables the player to interact with the game world, move around the map, and perform various activities, such as studying, eating, or engaging in recreational activities at various locations.

4. End of Day and Day Summary states: These states fulfill the user requirement for allowing the player to track their game progression ("UR_CLEAR_FEEDBACK") as well as ("FR_TRACK_STATISTICS"). At the end of each day, the player receives a summary of their actions and progress, enabling them to monitor their performance and make strategic decisions for future gameplay.

Interim versions of all the diagrams can be found at:
https://ivohadley1.github.io/architecture.html