Backlog CBL

Setting up the window (low priority)

- make it resizable

Demo:
After starting the app, resize the window. The aspect ratio will stay the same, while the resolution changes.
Note:
If the window is made really small, the game *might* become unplayable.

Making a menu (high priority)

Demo:
In (start → menu), test these buttons:

- play (goes to level selection system)
- settings (goes to a menu where you can change e.g. controls or volume)
- help (goes to a menu that says “nuh uh, you can figure it out yourself)
- exit (quits the app)
- back (for every back button, test if it takes you to the correct (previous) screen)

Note:
The listed items would be buttons available on the menu. Every sub-menu, and every menu in the project will have a “back” button in the top left to go back to the previous menu. Escape will work the same.

Making a level selection system (extra: character selection) (high priority)

- Level buttons
- (extra: characters)

Demo:
In (start → menu → levels), test if every level button takes you to its corresponding level. If character selection is added, that would be a different button that takes you to the different characters to pick from.
Note:
The character selection menu will be added but take you to an empty screen until they are implemented.

Make a pause menu (similar to menu) (low priority)

Demo:

In (start → menu → levels → any level), click the assigned button or escape to pause the game and freeze the game engine. Click on the buttons that appear after you click the pause button. Test these buttons:

- Exit level (goes to level selection menu)
- Resume game (resumes the game engine)
- Settings (goes to the settings)

Note:

This menu will *not* have a back button, since it already has replacements.

Player controls (high priority)

- simple movement buttons (go left, go right and jump)
- double jump (optional)
- dash (optional)
- attack (optional)
- crouch (optional)
- hanging on ceiling (optional)

Demo:

In (start → menu → levels → any level), test the movement buttons for the player.

Note:

The controls are initially set to “a”, “d” and “space” but can be changed in the settings menu. If characters are added, the double jump and dash would be assigned to certain characters.

Implement graphics (high priority)

- static camera
- simple sprites for character(s), map design and interactables

Demo:

In (start → menu → levels → any level), show the game with a bit more of a visually pleasing look.

Note:

The camera is static, which means that the character moves across the screen. At first, the graphics will be really basic to make the game work, actual sprites will be added later to make it more enjoyable and give the game a style.

Implement physics (high priority)

- hitboxes (for the platforms, character(s) and enemies)
- gravity
- enemies (optional)

Demo:
In (start → menu → levels → any level), test if the hitboxes work; the character does not fall through the map. Test if the character falls at all to prove the existence of gravity.
Note:
Testing hitboxes consist of: test if the player does not go through a platform that has a hitbox.

Interactions (lower priority)

- finish level (door/requirement/key)
- enemies (attack/get attacked)
- npcs (conversation/shop)
- coins (pickup/use in shop)

Demo:
In (start → menu → levels → any level), test if the player can interact with interactable things correctly.
Note:
These last three topics (Interactions, Sounds and Animations) have lower priority, meaning that they will only be added if there is enough time to do so.

Sounds (even lower priority)

- pling (coin pickup or something positive)
- auwie! (getting hurt)
- yippee (yippee)
- pieuw (attack)
- hup (jump)
- waaaAAAaaAA (scream)

Demo:
In (start → menu → levels → any level), test if the sounds play at the right moment.

Animation (very low priority)

- walk
- jump
- attack (extra)

Demo:
In (start → menu → levels → any level), test if all the animations play at the right inputs.

Topics of Choice:

Setting up GitHub for version control (1st topic of choice)

- make a branch for both of us and a backup

Demo:
Push two different versions of the same program and have it merge them properly.

Documentation GitHub

We started off with some difficulties, the repository should not be cloned into a OneDrive, because it would be very difficult to work with. Our experience using GitHub is very positive and we learned a lot about GitHub.

Link:
See README file.

Game design (2nd topic of choice)

- research about what makes a game fun
- decide what features to add to make the game fun

Demo:
Showcase the chosen features and document them.
Note:
These features will probably be added throughout making the game, so not everything can be demoed right away.

Documentation Game design

Platform video games are a very popular genre amongst the genres of video games. It is also a very old genre, going all the way back to the early 1980s, with the creation of the two old arcade machine games: Donkey Kong (Nintendo, 1981) and Space Panic (Universal, CBS Electronics, 1980). Since then the platform video game genre has really evolved and gotten more complex. Like the games Super Mario Odyssey in 3D and Celeste in 2D. With all these new games, there is a question: What makes a good platform game?

An element is control. That means to test the player’s skill with precise movement with hard obstacles that require precision. Each game has its own way of making that. But in the limited time we are given for this project it is not that sufficient to implement this without getting time stress.

Another element is puzzles. Puzzles make a platform game have more depth and it also makes the game more challenging for the player. You have a lot of different puzzles like the 2-player puzzle games. In those games you will have to help each other get to the end of the level. There are much more variants of puzzles in platform games. But implementing a puzzle in a platform game takes a lot of time we will not be implementing this in our game

An element that we could implement in our game is mechanics. Mechanics are a fundamental element to make a good platform game and there are many types of mechanics. Some mechanics are easier to implement like double jump and dash. And others are harder to implement and more complex like the Mario’s hat in mario odyssey.

So it will be easier for us to implement mechanics, because the easier mechanics do not cost that much time to implement and they are not that difficult to code. We are gonna implement a mechanic that lets you hang on a ceiling, because we want something not that hard but also a little bit challenging.