# TODO

## Tasks

* Make GitHub repo private
* Investigate how to share assets between applications by exporting/importing custom Unity packages (WM)
* Investigate needed Unity license and pricing for commercial use.
* Figure out how to setup Unity editor to use degrees for angles.
* Implement HUD UI to enable quick teleportation to arbitrary project/POI
  + Separate ActivateProject/TeleportToPOI menu, or combine in one menu?
  + Show preview thumbnails for projects?
  + Show preview thumbnails for POI?
  + Group POI buttons per ‘building layer’?
    - Surroundings
    - Floor X
    - Ground floor
    - Basement
* Network menu: Show preview of chosen avatar.
* Implement ‘Laser pointer’ functionality
  + Implement local:
    - Walkthrough mode: If user presses R index trigger, show R pick ray.
    - Implement passing laser pointer related state to server
      * Add to AvatarState
    - Implement showing laser pointer for remote avatars
      * Add L and R pickray objects to Avatar prefabs?
      * Update L an R pickrays for remote avatars from their received AvatarState.

## Bugfixes

* Fix avatar hand issues
  + location.
  + Fix hand model Z-buffer/face orientation issues, or find/use better model.
* Implement optimalisation for communication between Server and Client on the server host: do not use network to communicate with the local client.
  + Implement IClientConnection interface with concrete implementations RemoteClientConnection and LocalClientConnection?
* **IF possible:** Fix the necessity in the server to Binary-parse-to command all messages before propagating them (Implement concept of messageType ClientOnly/CLientServer/ServerOnly?)
* Network menu: Implement transitioning between NetworkModes during an application session.
  + Implement transition Server -> Standalone:
    - Shutdown server
      * Broadcast a DisconnectServerCommand upon
  + Implement transition Standalone -> Client:
    - initialize Client
    - connect Client to Server
  + Implement transition Client -> Standalone
    - Shutdown cient
      * Send DisconnectClientCommand to Server.
* Project KS046:
  + Model
    - L0 Technische ruimte: Add: boiler, ketel
    - L1 Slaapkamer 2: Add: Buro stoel.
    - L1 Slaapkamer 1, 3: Add: Nachttafeltje
    - Add eetkamer and leefruimte lamp plafond
    - Fix positie L1 bed 2
    - Fix groef terras achter, en grasplooi rechtsachter.
  + POI
    - L1 WC: in muur?
* Only show button mapping UI when controller height is below certain treshold irt center eye anchor.
* Fix FadeOut/FadeIn: Add to separate (render)layer ‘FadeLayer’. Fade layer should be rendered as last, without Z-testing.(flickers + depth-clipping) -> moved back from viewer a bit
* Implement Teleport area concept
  + Implement ‘MoveToTeleportAreaWarning’ UI
    - World-scale UI canvas
      * anchored to center eye
      * By default invisible.
  + Implement TeleportArea GameObject
    - Cylinder primitive with semi-transparant material.
    - By default invisible
  + Implement showing/hiding TeleportArea and warning
    - Set visible upon TeleportCommand when user is not in teleport area.P
    - Set warning to invisible:
      * When entering teleport area
    - Set teleport area to invisible when teleporting.
* Implement option ‘EnableTeleportAreaDebugging’ to aid in debugging. Flags whether Teleport area requirements are enforced when running in editor:
  + If ON, user can teleport only from whitin teleport area.
  + If OFF, user can teleport from outside of teleport area.
* Performance improvement:
  + Implement LayerConnection concept
  + Implement auto-generation of LayerConnections from found groups in model upon loading project? (this will be the ‘Walls topside’ groups)
  + Implement only show LayerConnections when necessary. (source layer is visible, target layer is not.
* Implement hiding furniture in layers above/below center eye anchor.

## Design Defects

* Remove Left/Right ControllerAnchor, and use Left/Right TrackedRemote instead.
* Improve code related to controllerstate and ButtonMappingUI
  + in order to mitigate code verbosity/duplication.
* Implement ButtonState concept (Default/Down/Pressed/Up)
* Implement ‘Command’ class?
  + CommandResetViewerLocation (click r thumbstick)
  + CommandTranslateModel
  + CommandRotateModel
  + CommandResetModelLocation (click l thumbstick)

## Features

### Buddy space

* Implement application state 'budy space definition'
* Show ‘passthrough’ camera feed (OVR API does not allow this though! ☹)
* Show Pickray
* Let user define 'shared buddy reference system':
  + Make user define first axis of ref system:
    - Pick 2 points/markers (on ground level plane?) in opposing corners of the play area using pick ray
  + Together with vertical axis, and axis orthogonal on previous 2 axes, the ref system is defined.
* Implement broadcasting own avatar position expressed irt buddy shared reference.
* Implement updating locations for remote user avatars using the locations irt buddy shared reference.

### Doors

* Implement Model Doors manager
* Implement gathering doors from model upon loading a project.
* Implement picking doors to open/close them.

### Building lights

* Implement toggling building lights on/off

### Environment

* Implement setting sun position: hour-of-day , sun azimuth
* Implement setting animation speed for time (sun position)
* Implement weather system
* (Fog, clouds, rain, snow...)
* Implement setting weather

### Furniture

* Implement option to toggle global furniture visibility.
* Implement a way to toggle visibility of furniture by picking?
* --> How to unhide???
* Implement custom(prettier) skyboxes

### Vegetation

* Compose a vegetation model collection
* trees
* hedges
* bushes
* grass textures
* Implement dynamic vegetation
* Implement option to make static (if needed for performance reasons).

### GFX settings

* Implement menu option to enable/disable reflection
* Implement a way to gather all geometry that needs reflection shader
  + Mirrors
  + Glass
  + ...?
* Implement reflection shader
* Apply reflection shader on appropriate geometry.

### Edit Mode

In Edit mode, user can edit the active project

* POI: create, edit, delete
* Furniture: create, edit, delete
* ...?

Will we even do this? Not usefull without making editable stuff transferable between builds which requires additional development… Only do when we have a workflow that needs it! For the time being, all project editing will be don in Unity editor, which suffices for now.

# DONE

* Implement OVR driven input.
* Implemented Unity Input (buttons and kb).
* Fix texture load from SKP files. (No spaces in texture names allowed, material name must == texture name.)
* Implement loading projects from separate scenes asynchronously
* Implement aplication states:
* Default: While previewing the active project. Input is active. Model is visible
* Teleporting: While teleporting to another Project and / or POI. All input is blocked (in order to prevent requests to activate Project or POI),....
* First fade out
* Then show title of new project while async loading it
* Then fade in again
* Implement immersion modes: Walkthrough and Maquette
* Set menuMode default to ‘None’
* Fine-tune size and font for Controller-anchored menus.
* Walkthrough mode: Implement showing the name of active POI in RIght COntroller Menu.
* Implement showing projec name in Left controller Menu
* Implement fading (out, in) transition when teleporting (improves user confort).
* Implement pickray.
* Maquette Mode: Implement hiding model layers using picking.
* Unhide all model layers when exiting immersion mode ‘Maquette’
* Editor mode: Implement showing keys next to button action label
* Implement PickRay class
* Generate some bumpmap gevelsteen textures from DAAS, and add them to resources repo.
* ImmersionModeWalkthrough: Implement exclusive lock between rotateTrackingSpace() and MoveUpDown()
* Maquette immersion mode: Implement exclusive rotate/translate locking mechanism: While rotating, no translation can be performed and vice-versa.
* ButtomMappingUI: Implement deadzone for thumbsticks.
* Implement dynamic pickray color:
* If no hit: Use textured cylinder with fading out (fully opaque -> fully transparent) uniform white color
* If hit: Change to red color.
* Implement getting build datetimestamp for composing build version number automatically.
* Fix HUD menu not working
* ImmersionModeWalkthrough: Implement a way to align model with tracking space:
* LT left and right: rotate model around center eye position.
* RT up, down, left and right: translate model (relative to viewer)
* Make PickRay prefab
* Instantiate PickRay prefab under both hands as R PickRay and L PickRay
* Disable L pickray, use R pickray only
* ImmersionModeMauette: Use R index trigger to toggle model layer visibility.
* Implement FPS counter HUD control (Simple text to bottom of view.)
* Performance improvement: Enable backface culling on model (set sketchup import setting)
* Implement GFX debug mode with HUD UI. (show gfx quality +fps)
* Generated alfa maps for wall and floor tiles, and use them in unity.
* Fixed lighting issues: updated HOWTO.
* ImmersionModeMaquette: Make model rotate around ModelANchor possition(center of the building), if present.
* Implement connecting to buddies using TCP-IP
* Server/Client: Implemented sending arbitrary .net objects by wrapping them in a Message.
* Implemented CommandTeleport
* Implemented CommandSetImmersionMode
* Implemented HUD menus as separate Panels with custom behaviors.
* Fix avatar default head rotation: Rotate around Y with 180 degrees.
* Implemented picking menu UI in VR mode
* Implemented menu UI 'Selected' color as yellow to improve working in menus in Editor mode.
* Multiplayer network modes: Fix avatar being spawned for local client.
* Multiplayer network modes: Fix avatars for clients to be of the type defined by the client.
* Implement ConnectClientCommand.
* Implement DisconnectClientCommand.
* Implement ServerShutdownCommand.
* Network menu: Implement sending a SetClientAvatar upon selecting avatar.
* Fix avatar positioning: 1) Also send hand positions. 2) Rename ‘position’ and ‘rotation’ to ‘headposition’ and ‘heatRotation’ 3) Use head position and rotation to update head location 4) Offset avatar body from head position by Y -0.8.
* Implement UI interactions in HUD menu using pickray.
* Walkthrough immersion mode: Implement exclusive tracking space rotate/translate locking mechanism: While rotating, no translation can be performed and vice-versa.
* ImmersionModeWalkthrough: Show Boundary while translating tracking volume.