Troops Training

Step 1:

Clicking Barrack and clicking training icon:

Then GlobalUIManager analyse click and check:

 else if(ClickedObject.GetComponentInParent<BuildingInstance>()){

            Debug.Log("building clicked.");

            ClickedObject.GetComponentInParent<BuildingInstance>().BuildingClicked();        }

//this will enable icons for that building.

Now clicking Training icon trigger a function in TheBarrack.

public void TrainingIsClicked(){

        //triggered by clicking UI button train.

        troopsTrainingManager.TrainingIsChosen(gameObject.GetComponent<TheBarrack>());

    }

Then in TroopsTrainingManager:

public void TrainingIsChosen(TheBarrack TheBarrack){

        theBarrack=TheBarrack;

        troopType=theBarrack.barrackType.ToString();

        if(IsBarrackOccupied()){

            uITroopsTrainingManager.TriggerUIForOngoingTraining();//some debug messages.

        }

        else{

             uITroopsTrainingManager.TriggerUIForTraining();

        }

    }

In UItroopstrainingmanager:

public void TriggerUIForTraining(){

        //levels, cost , barracklimits

        //update barrackCapacity UI

        barrackCapacity=troopsTrainingManager.GetTroopsCapacity();

        //pass it to slider

        troopsTrainingSlider.SetMaxBarrackCapacityForUI(barrackCapacity);

        //display total troops capacity and type ++++++

        barrackType=troopsTrainingManager.troopType;   //gives types

        troopsTypeUI.text="Train "+barrackType.ToString();

        //triggering Starting training ui

        StartingTrainingUIPanel.SetActive(true);

    }

Now in TroopsTrainingManager:

 public int GetTroopsCapacity(){

        //this will get troops training Capacity of the that given barrack

        //this will be called by ui manager

        return theBarrack.TrainingCappacity;

    }

Now troopsTrainingSlider:

//this will set the max for each slider limit for ui.

public void SetMaxBarrackCapacityForUI(int barrackCapacity){

        //this will be called by ui manager

        BarrackCapacity=barrackCapacity;

        level1Slider.maxValue =  BarrackCapacity;

        level2Slider.maxValue = BarrackCapacity;

        level3Slider.maxValue = BarrackCapacity;

        level4Slider.maxValue = BarrackCapacity;

        level5Slider.maxValue = BarrackCapacity;

    }

This will open a panel with 5 slider with their max limit set to barrack training capacity.

Summary:

Clicking barrack ,global ui check if Thebarrack is in parent .if found ,it call it’s buildingInstance function to enable it’s icon. Clicking training icon calls the barrack function which sends it’s GO to training manager,which stores that GO .it first check if it is occupied or not. If it is not occupied ,it gets barrack troops capacity and it’s type set it to ui panel .which limits max values of each according to barrack capacity.

//there is a slider mech which is too complex so we skip to training click

Step 2: train is clicked.

Clicking train triggers a function in UITroopsTrainingManager

public void TrainIsClicked(){

        troopsData=troopsTrainingSlider.ReturnTroopsData();

        if(tradingManager.IsEnoughResource(trainingCost[0],trainingCost[1],trainingCost[2])){//w,g,s,t//this is simple.

            tradingManager.SpendingResources(trainingCost[0],trainingCost[1],trainingCost[2]);

            troopsTrainingManager.StartTrainingProcess(troopsData,trainingCost[3]); //time

            EndStage();//refresh global permission

            RefreshUI();//disable training panel

        }

        else{

            Debug.Log("Not Enough");

        }

    }

//this will get all the slider values and return as a int array .

 public int[] ReturnTroopsData()

    {

        UpdateAllValues();

        int[] troopsData = new int[5];

        troopsData[0] = level1CounterLM;  // Troops for Level 1

        troopsData[1] = level2CounterLM;  // Troops for Level 2

        troopsData[2] = level3CounterLM;  // Troops for Level 3

        troopsData[3] = level4CounterLM;  // Troops for Level 4

        troopsData[4] = level5CounterLM;  // Troops for Level 5

        return troopsData;  // Returns the array containing troops per level

    }

//Like :[11,21,33,41,52]

 public void StartTrainingProcess(int[] troopsData,int time){

        //get that barrack assign it.

        theBarrack.StartTraining(troopsData,time);

    }

public void StartTraining(int[] troopsData,int time){

        troopsDataLocal=troopsData;

        UpdateStateOfBarrack(true);//set barrack to occupied.

        //call function  for training

        trainingHandler.StartTraining(time);

    }

public void StartTraining(int time)

    {

        if (trainingCoroutine != null)

        {

            StopCoroutine(trainingCoroutine); // If there's already a coroutine running, stop it

        }

        trainingCoroutine = StartCoroutine(TrainingRoutine(time)); // Start the coroutine with the training time

    }

    // Coroutine that runs for the duration of the training process

    private IEnumerator TrainingRoutine(int time)

    {

        Debug.Log("Training started...");

        // Wait for the specified time (in seconds) before proceeding

        yield return new WaitForSeconds(time);

        Debug.Log("Training completed!");

        // Once the training is done, update the state of the barrack (or any other action you want)

        UpdateStateOfBarrack11();

    }

    void UpdateStateOfBarrack11(){

        theBarrack.TrainingEnded();

    }

 public void TrainingEnded(){

        //this is called by training handler

        //allow the icons here

        Debug.Log("enable icon here");

        UpdateStateOfBarrack(false);

        //tell ui training ended or triiger message

        //for now add troops directly

        troopsCountManager.UpdateTroopsCount(barrackType.ToString(),troopsDataLocal);

    }

public void UpdateTroopsCount(string barrackType,int[] troopsData){

      //this will be called by barrack for now.

      //when training is done

      // Make sure that troopsData length matches the size of your arrays (e.g., 5)

    // Check for the barrack type and update the corresponding array

    if (barrackType == "Cavalry")

    {

        for (int i = 0; i < cavalry.Length; i++)

        {

            cavalry[i] += troopsData[i];  // Add corresponding troop data to cavalry array

        }

    }

    else if (barrackType == "Infantry")

    ……

    else

    {

        Debug.LogError("Unknown barrack type in countmanager: " + barrackType);

    }

    Debug.Log("Message should be displayed mentioning troops detailes which were added.");

   }