**ABEEHA’S FINDINGS**

**ABDULLAH’S FINDINGS**

**GD S1: The world of HCGs along with different genres**

Most gaming revenue in mobile, fastest growth in market size (see pic)

Mobile gaming dominating atm and continuing to take big share of pie (2021 stats)

Many stats given as to why mobile gaming spread

HCGs = ‘Super fast food’

Simple to understand – anybody can play

Try to reach to as many audiences as possible

Ascending order of complexity and sessions’ length:

HCG < Casual (slightly more complex) < Midcore (multiplayer, complex interactions) < Hardcore () [ see image if needed ]

HCG Elements:

1. Simple, short, satisfying
2. Fun for everyone (get away from ‘gamer’ label)
3. A loyal following TO THE CATEGORY, NOT THE GAME

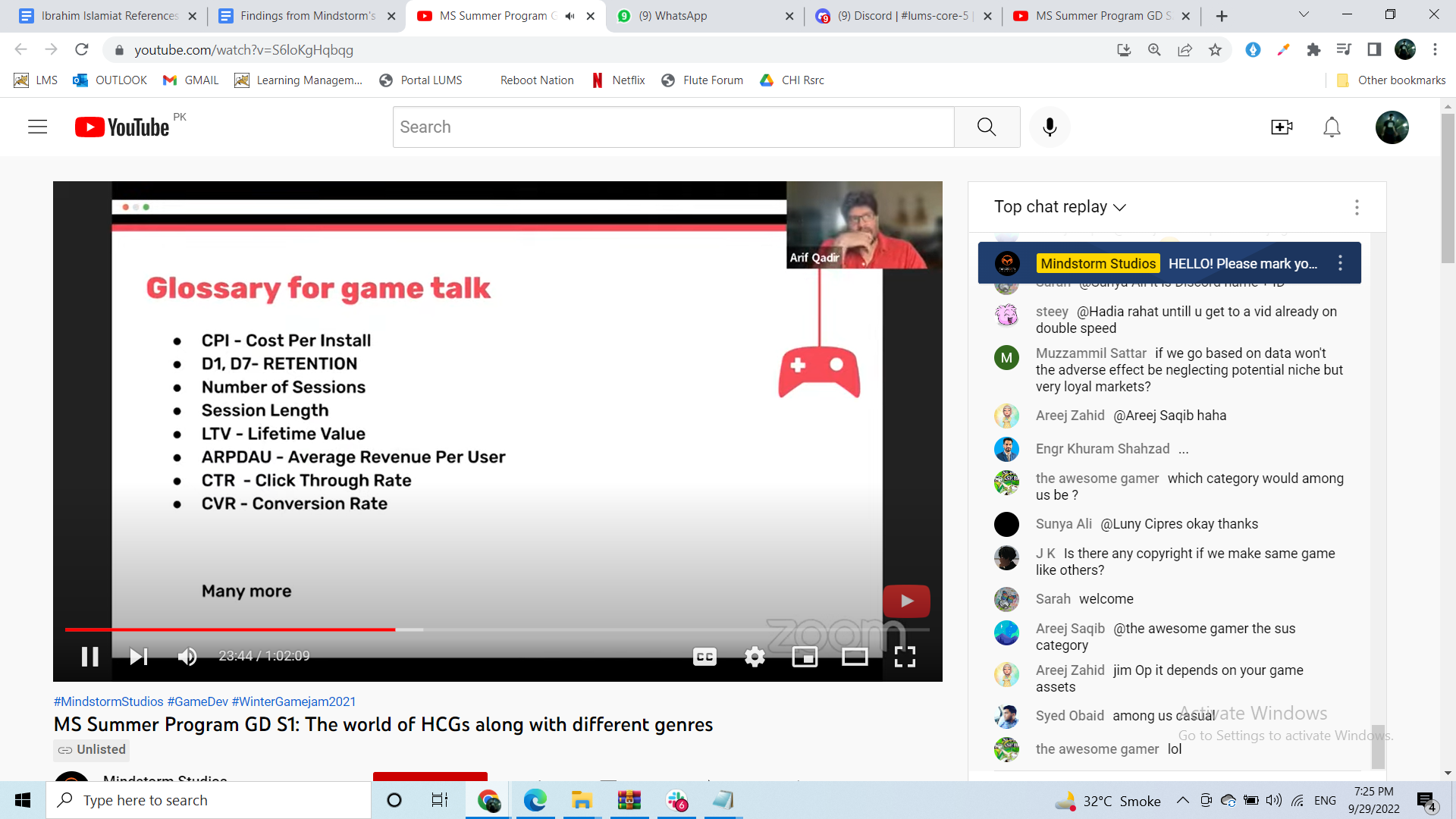
→(gamer carries on to next game after few days)

1. A new way to monetise (ads high frequency and not that disrupting)
2. Driving down CPIs (cost per install)
3. Speed and momentum paramount

→ soft launches for testing, iterating, optimizing

→ produce and iterate fast

1. Data obsessed



LTV – how long user keeps game installed

ARPDAU – every day how much revenue per user

CTR – how many people clicked ad for your game

CVR – how many people from clickers installed your game

HCG games’ users cost less and give you less compared to casual. Gotta get it to scale for HCG to be profitable.

Quick narrative.

No/ fake multiplayer in HCG

Some Mechanics: pushing, swerving, forward-running, draw-and-puzzle, dexterity, grow, tapping, merging (check around 34:00)

→ not much depth in mechanics

People wanna have fun…e.g. Puzzles mein haar hi nhi sakte. Bit of ASMR effect too.

QNA:

Subway surfers not HCG

Clash of Clans Midcore, long playtime + constant updates needed

Mechanics are key to create depth with limited controls? More challenging to do so.

Profits in cents so need massive # downloads

All genres…all categories to cater to bigger audience. Don’t go into niches e.g. goth girl game.

Core loop in HCG: collecting different items, powering up character

→ Based around core mechanic, interact with it. It gives you reward in turn. Repeat.

**GD S2: Game Design 101**

GENERAL GAME DESIGN

Game Design

→ what goes into game, deciding pillars

/= graphic designing

Game Designer

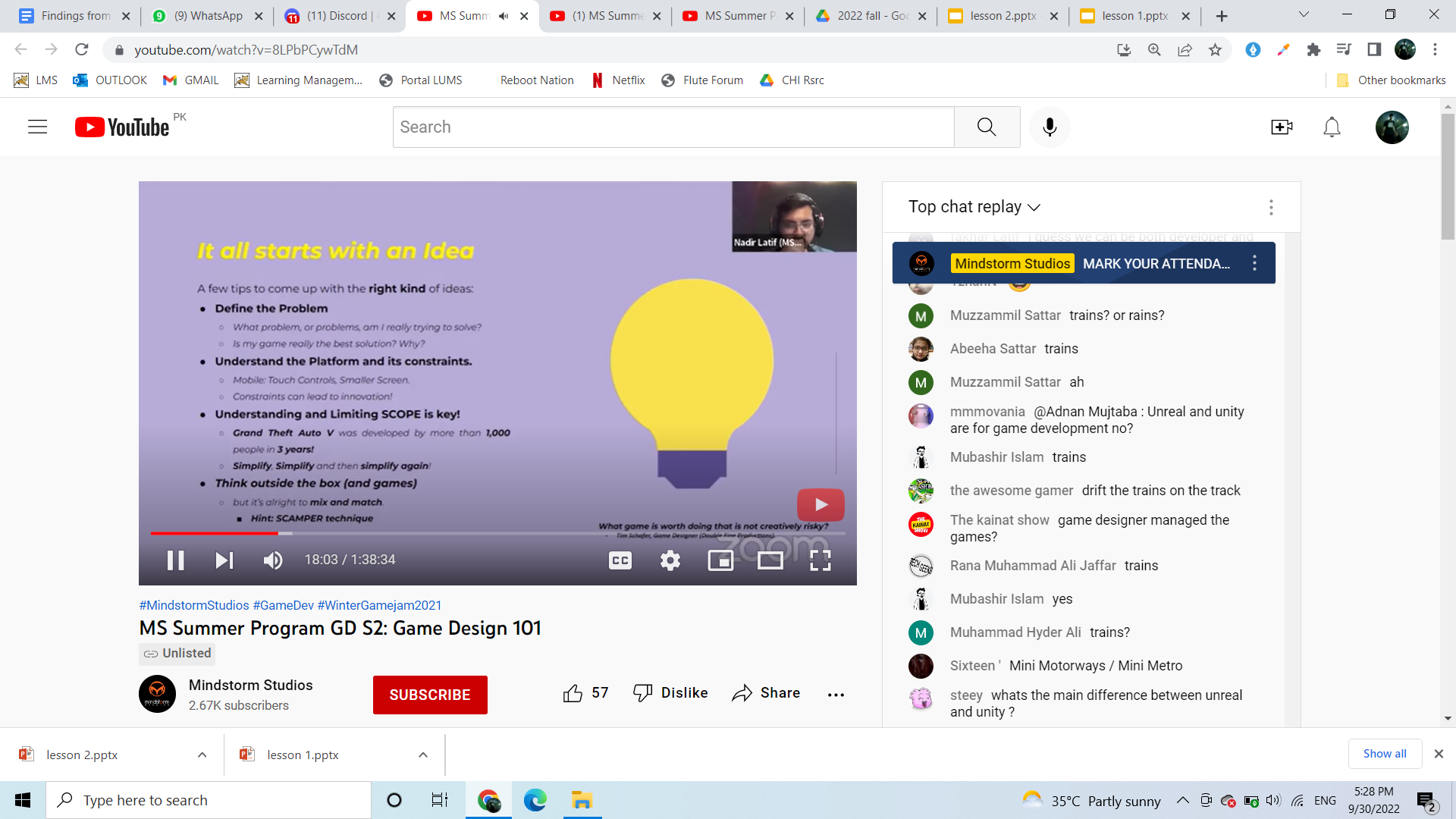
→ ideas, conceptualizing and documenting features of game, analyses players’ behavior

→ further subclasses too like combat designer, gameplay designer etc

→ involved throughout the process + multidisciplinary role

Ideation

→ think of your game as trying to solve a problem!



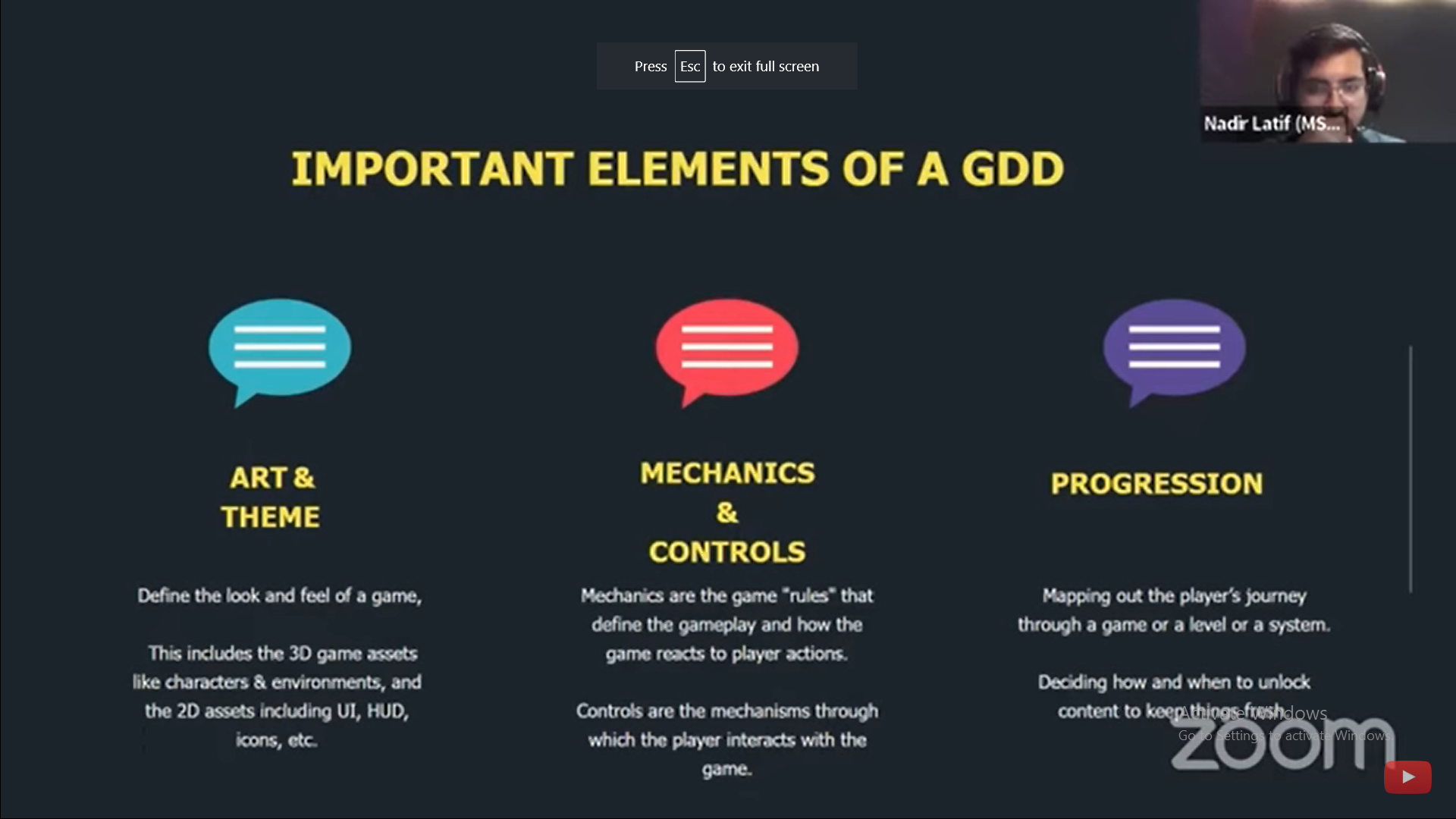
→ simplify & repeat…is base level fun or not? E.g. Mario

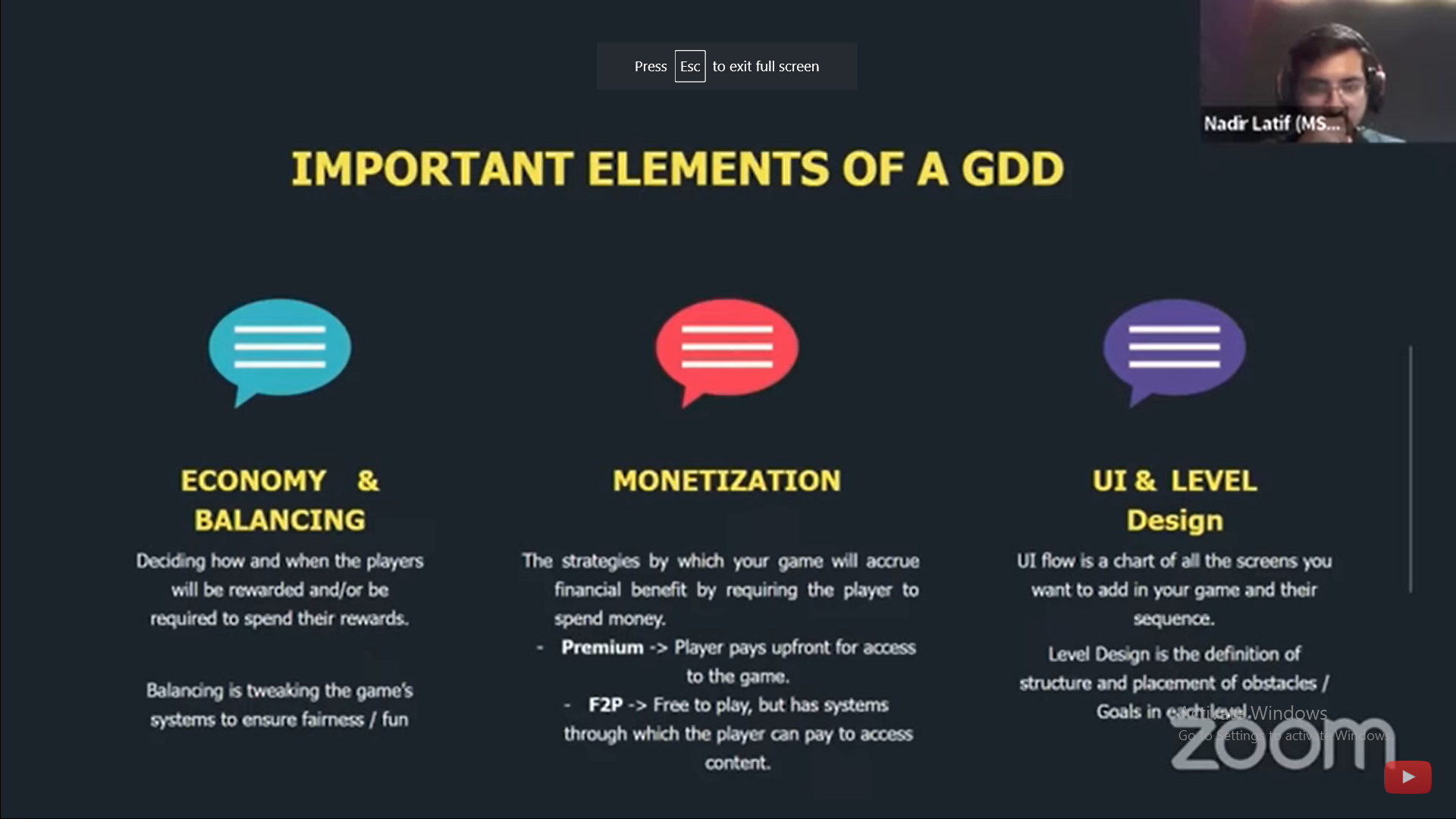
GDD Game Design Document

→ living document that’s blueprint (describes UI, characters, gameplay etc) throughout production

→ divided into:

* Art & Theme
* Mechanics & Controls
* Progression





HCG GAME DESIGN

→ short session length + fun game mechanics

Super simple gameplay loop

Easy to get in and out

Gratifying gameplay

Simple meta/economy

HCG Controls: tap, swipe, drag/swerve, virtual joystick

(can mix and match) Progression: level based, performance based, upgrade based (buy new sword get new enemies)

Usually multiplayer in HCG are fake bots

Mechanics: rising/falling, timing, running, stacking, merge, puzzle, idle, arcade idle, simulation, growing, drawing (52:00 starting)

→ Mechanics always evolving

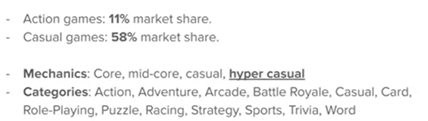
Monetization: ad banners, rewarded videos, interstitial ads, in-app purchases

Right game for the right audience

**GD S3: Data Driven Game Design**

→ use data to facilitate game design

**GD S4: Game Publishing, Business Models and Managing Store**



Business models: F2P Apps, Premium Paid Apps, Recurring Subscription Apps. Contain:

→ In-app purchases (IAP), Ads,

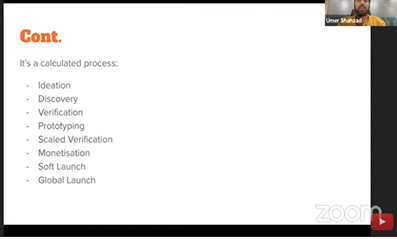
→ usually free to download, bonuses are paid for

Ads:

* Banner (less money, too small)
* Interstitial (more money, but can still be skipped)
* Rewarded (most money, but frequency less as non-compulsory)

1 to 1 relationship between developers and publishers → different specializations now

(finance, manage releases, marketing, expert in data analysis)



Above slide

 around 35:00

20-50 ideas on weekly basis usually in software houses

Verification: fake gameplay ad, see how many people interact with it as cheap verification

Prototyping: 5 stages usually. Pick most important elements and test their attractiveness in market.

Scaled verification: players’ engagement with real app on playstore → more budget, content, levels and features for bigger audience

Soft launch: launch game in few countries to gauge response and cost

Global launch: push game globally

Managing Playstores

Different analytic metrics and stats

QNA

Ad displayed → little money

User clicks → some money

User downloads → more money

* Constant experimentation goes on with # ads and # session length, to optimize both. Set a minimum benchmark for user’s session length.

* Most popular are Google Playstore And Apple’s App Store
* Google Playstore: 25$ lifetime
* “Pakistan mein game bana rhay ho to trend pr bana rhay hogay”
* You don’t choose what ads go where. Some Ads Center does that
* If user has ad blocker or no wifi, game doesn’t generate money for the ad

**AMAAN’S FINDINGS**

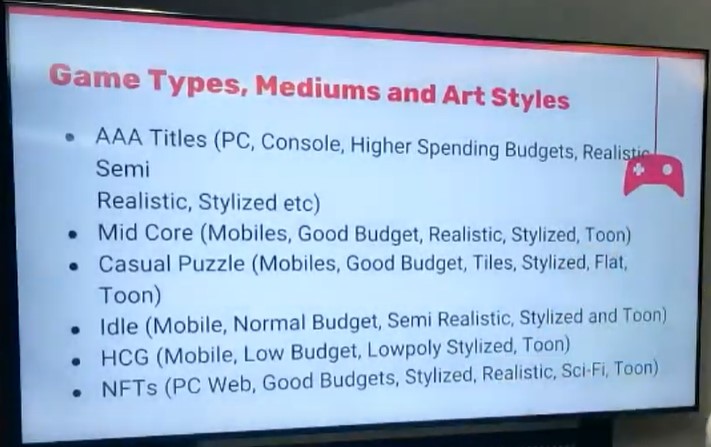
**GA session 1:**

**- Industry started in 70s**

**- 8bit graphics to advancements**

**- Art styles and genres which uplifted the standards of users**

**- 75% of gamers say visual plays an important role**

****

**- Art style is visual appearance.**

**- Stylized means semi-realistic (Tekken characters powers and body)**

**- Mid core (mobile games)**

**- Dimensions are important in 2d and 3d**

**- 3d- every angle of an object**

**- 2.5D – 3d character in 2d space or vice versa**

**- 2d- animate or static**

**- 3d- first model then animates by artist - object – model gets in engine and then animate or static**

**- 2.5d- decides an angle (blender) 45 let’s say 45 , image by , frame rate important 4 images per frame is our idle, 12 (180/15) \*4 spreadsheet, many pics attached to one image**

**- Less angles enough if works**

**- Minimalistic, noise free and to the point interface screens**

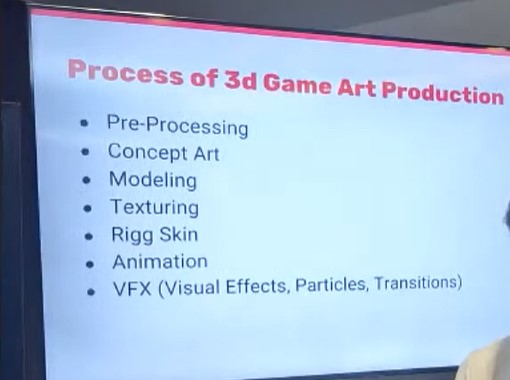
**- Time to time visual quality check of production and improvement feedback**

**- Thing to look: Lighting, camera, composition, scene setups, VFX and transitions**

**- Sketch and then model and look at UI/UX**

**- Use assets if it goes with your project else if unique make own**

**- 2d animator- make things alive, artist - sketch**

****

**- Decide everything about game (pre-processing)**

**- Sketches and modeling (model are gray)**

**- Texture is look and feel**

**- Rigg skin and animation can be imported (limits of animation like bones)**

**- System one thinking is getting to know things from your observation like any child**

**- First 3 secs should convey what is gameplay (single mechanic swipe or tap) HCG games**

**- Idea can be from anything youtube or real life**

**- Minimalistic design Screen :- top: UI hub, middle: main character or hurdle, bottom: controls**

**- Color psychology: generic: blue, green**

**- Endless game can’t have human eye angle (flaw) hurdles will be not clear**

**- Ads are ok bcz user know about progress**

**- Exercise (critic on game pics) -lighting -light angle -shadow lines -**

**GA Session 5:**

**- UI trends over time**

**- From glossy and gradients to flat design (curve sense)**

**- Then long shadows in icons**

**- O0H depth (flat and simple)**

**- Neomorphism (soft and same colors)**

**- Glass morphism (glass feel, blur at back, no glossy colors)**

**- Neobrutelism (incomplete feel, less used)**

**- Neon colors**

**- 3d elements**

**- Retro UI (bold colors, typography, soft gradients)**

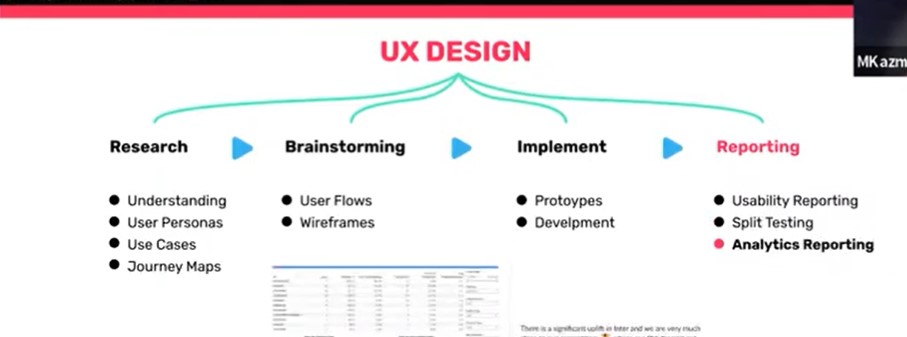
**- UI/UX difference**

****

**- Same button family**

**- UI:- device dimension, layout, negative space/spacing/padding/ICONS or buttons, typography, colors/contrast, motion design**

**- UX: research, brainstorming, implementation, reporting**

****

**- Test with different sets of features**

**- Exercise: making of UI**

**BAKHTIAR’S FINDINGS**

GP S1:

What is a Gaming Engine?

It is a software framework for designing and developing games. Many game engines available in the market: Unreal Engine by Epic Games, Unity, Godot. Unity is very popular because of extensive community support and regular updates. Games made in unity include Super Mario Run, Flappy Bird and Subway Surfers.

Unity

Unity is a cross-platform game engine. Games that can be created are 2D, 3D, VR and AR. It is also used in other areas such as film, architecture and engineering.

Why Unity?

It is easy to use and its scripting language is C#. Its free version covers most of the use cases. It has a rich Asset Store where ready-made assets can be downloaded and added to the game. It has online documentations, tutorials and a helpful community.

Basic Components

Scene

Gameobjects

Components

Prefabs (pre-configured reusable GameObjects that you create in the scene and store in the project)

UI (like the text, buttons that appear on the screen)

Media Assets like png, mp3, textfiles etc

Scripting

Default Visual Studio IDE

Paid editors like Rider

Visual scripting in Bolt

Unity has a predefined life cycle.

Interface

We can create any predefined object such as cube by right clicking in the files panel and going to 3D and then Cube.

We can add components such as Box Collider( determines the collision of the object with other objects), Rigid Body and script by clicking on the Add Component button in the lower left corner of the screen.

To create a script, in the assets menu, right click and click on create and then C# script.

Unity Life Cycle

Initialization phase

Physics phase

Input Phase

Game Logic Phase

Animations

It has a rich animation system (sometimes referred to as ‘mecanim’)

Easy workflow of animations using its window.

Support from imported animation clips and clips made in Unity

Animating different body parts using logic

Ability to apply animations from one character model to another.

Powerful tweens plugins like Dotween, Itween

To start animation, we have to add animation to the animation controller. We can switch between different animations using conditions , for e.g. if we want to switch from crouched to sprint, I will click on a button which will set a variable isSprinting to true hence using a script we can start the sprinting animation based on the conditioning.

Physics

Uses Nvidia PhysX

Support for Havok Physics

Rigid Body

Colliders

Joints

Ragdoll Physics

UI

Canvas

Canvas Render Modes(Overlay, World Space)

Components like images, texts

Animations using animation controller and tween

AutoLayouts(Grid, Horizontal, Vertical)

Profiler

Gets performance information about the application

Pinpoints issues related to performance

Highlights multiple issues like CPU, GPU, rendering, memory, physics

Audio

Unity has support for playing sounds in 3D space

Supported formats: mp3, wav, ogg

Audio Mixer

Multiplayer (backend)

Firebase

Photon

AWS

Azure

**HAMZA’S FINDINGS**

**GP S5: Arts and Animations**

**(Tutorial on how to add animations to an object)**

**Animations:**

Help us

* To tell stories
* Communicate ideas
* Add life to objects

**Animator Controller:**

Controls all the animations of the object/character.

**How to open Animator Controller in Unity:**

Add component of the animator from the right side of the Unity.

**Note:**

Visit Mixamo website to download animations for/according to your character.

<https://www.mixamo.com/>

To add animations, open the animator window from the *Window* menu.

**QNA:**

Q: Is it possible to make changes to the animation downloaded from asset store or maximo?

Ans: Yes, it is possible. Open the animations window (*Window —> Animations*)

**Animator Window:**

* Add states/animations and then transitions from one animation to another (transitions create the order of animations).
* **Any state:** It is used if we want to start any animation (after the prev animation ends) based on some logic/condition.
  + Set the parameters in the parameters tab after using *any state* in the animator window.
  + Set the logic/triggers in C# script file.

**DOTween:**

DOTween is an animation engine for unity.

**For instance:** if we want to add jump, move, rotate, etc. animation to an object, DOTween library is used.

**Particle System:**

In order to add smoke, flame, or some other similar effect, we need to use particle system feature in Unity.

Right Click in the hierarchy window —> Effects —> Particle System

**GP S6: Unity’s UI/UX Systems**

**Canvas:**

Canvas is game object with a canvas component on it and all the UI elements must be the children (inside) of such a canvas.

**Note:**

UI elements in the canvas must appear in the same order as they appear in the hierarchy.

**Render Modes:**

* Screen Space — Overlay (Used when you don’t want to use camera in your scene)
* Screen Space — Camera (Used when you want to use camera in your scene)
* World Space.

**Canvas Scaler:**

* Scale with screen size. (Resolution changes according to the screen resolution)
* Constant Pixel. (Resolution remains constant)

**Anchors:**

Anchors are used to stretch objects according to screen sizes.

**Visual Components:**

* If you want to use text, use *Text Mesh Pro* (with more features and is of high res) instead of simple *Text.*
* If you want to import some image from Photoshop and use in Unity. Import the image and change its *Texture Type* to Sprite (2D and UI).
* **Image Type** — **Sliced:** In order to change the size of the image while keeping the original form (aspect ratio) same, make the *Image Type* as *Sliced.*
* **Image Type** — **Filled:** It is used when we want to show processing, like when is loading, so some image is filled gradually showing progress.

**Note:**

In order to add any visual component, such as button, text, image,

Right click on the hierarchy window —> UI —> Text/Image/Button.