

Tien Le  
System Specification  
CSC 3150: Systems Design  
Professor Andy Cameron  
June 3, 2024



## FOCUS CRITTERS

## Table of Contents

<b>1. Executive Summary .....</b>	<b>3</b>
<b>2. Introduction .....</b>	<b>3</b>
<b>2.1. Problem Statement / Project Vision .....</b>	<b>3</b>
<b>2.2. System Capabilities .....</b>	<b>4</b>
<b>2.3. Non-functional Requirements and Design Constraints.....</b>	<b>4</b>
<b>2.4. System Evolution .....</b>	<b>5</b>
<b>2.5. Document Outline .....</b>	<b>6</b>
<b>3. Structural Model.....</b>	<b>7</b>
<b>3.1. Model Introduction .....</b>	<b>7</b>
<b>3.2. Class Diagrams .....</b>	<b>7</b>
<b>3.3. Metadata .....</b>	<b>8</b>
<b>4. Architecture Design .....</b>	<b>20</b>
<b>4.1. Architecture Overview .....</b>	<b>20</b>
<b>4.2. Infrastructure Model .....</b>	<b>21</b>
<b>4.2.1. Deployment Diagram 1 – Architecture Overview .....</b>	<b>21</b>
<b>4.2.2. Deployment Diagram 2 – Nodes and Artifacts.....</b>	<b>22</b>
<b>4.3. Hardware and Software Requirements .....</b>	<b>22</b>
<b>4.3.1. Hardware Components .....</b>	<b>22</b>
<b>4.3.2. Required Software Components .....</b>	<b>23</b>
<b>4.4. Security Plan .....</b>	<b>23</b>
<b>4.4.1. Security Overview .....</b>	<b>23</b>
<b>4.4.2. Security Plan .....</b>	<b>24</b>
<b>5. User-Interface .....</b>	<b>25</b>
<b>5.1. User-Interface Requirements and Constraints.....</b>	<b>25</b>
<b>5.2. Window/Screen Navigation Diagram .....</b>	<b>25</b>
<b>5.3. UI Wireframes.....</b>	<b>27</b>
<b>5.4. Reports: "Formal Output" Design .....</b>	<b>46</b>
<b>6. Appendices .....</b>	<b>46</b>
<b>6.1. Glossary .....</b>	<b>46</b>
<b>6.2. References / Bibliography .....</b>	<b>46</b>
<b>6.3. Supporting documentation .....</b>	<b>47</b>

## 1. Executive Summary

This system specification document provides a comprehensive overview of Focus Critters, an innovative mobile application designed to enhance productivity by combining focus sessions with engaging virtual pet rewards. The app helps users manage their time effectively by tracking focus sessions, providing detailed analytics, and offering motivational rewards as virtual pets.

Primarily intended for the developers building and launching the application, this document is a complete reference to Focus Critters' system architecture, object-oriented classes, and user interface designs. Developers should have experience in mobile application development, particularly with Android and iOS environments, and be adept at integrating cloud services. Familiarity with user experience principles is crucial, as the app aims for quick adoption and ease of use.

This document delves into the system's functional and non-functional requirements, class diagrams, deployment model, and security measures. It also includes recommendations for deployment strategies and visual design elements to ensure the app's success. The initial system proposal, complete with functional prototypes, is included to highlight the core features and vision of Focus Critters.

The end users of this application are individuals looking to improve their productivity and time management. Focus Critters app aims to provide a reliable and enjoyable solution by combining practical productivity tools with engaging virtual pet rewards. The document outlines the guidelines for the development team to proceed to the implementation phase, including robust system architecture and comprehensive security assessments to ensure a secure application.

In conclusion, this document details the design specifications required to achieve the project's vision. It thoroughly covers the system's architecture, class, deployment models, and user interface designs, providing essential guidelines for the development team.

## 2. Introduction

Focus Critters is a productivity app designed to help users improve their focus and manage their time effectively. Users start by setting a timer for their focus sessions, which can be private or group sessions. During these sessions, they can listen to music to enhance concentration. The app tracks the duration and productivity of each session, and users can view their session history and analytics reports to monitor their progress. As users accumulate experience points through completed sessions, they unlock new virtual pets as rewards. Being available on both iOS and Android platforms, the app ensures a comprehensive and accessible experience for users.

### 2.1. Problem Statement / Project Vision

Digital distractions like smartphones and social media make concentrating while studying or working difficult, especially for younger generations. Research shows that task performance decreases even when these devices are nearby. Focus Critters app aims to combat this challenge by embracing technology and creating a more engaging study experience. It leverages gamification through a virtual pet that thrives alongside focused study sessions. The app also integrates features proven to boost motivation, such as visual progress tracking and collaboration with friends. Users can personalize their study environment with music playlists, creating a more well-rounded and enjoyable experience that leads to better academic and professional outcomes.

Focus Critters app offers a win-win situation for various stakeholders. Students will gain a tool to improve focus, productivity, and academic performance while making studying more social and enjoyable. Working professionals can benefit similarly by enhancing focus and time management during

work sessions. Investors are presented with a commercially viable product with a diverse user base for potential revenue generation. The development team gains valuable experience building a successful application, and app stores benefit from distributing a high-quality app. Music streaming partners can expand their user base and gain valuable user data through integration with Focus Critters. Overall, this gamified study approach fosters focus, collaboration, and positive outcomes for all involved.

## 2.2. System Capabilities

Focus Critters include the following functional requirements, each corresponding to specific use cases detailed in Sections 4 and 5 of the System Proposal:

- Join a group focus session (UC-01): Users can join an existing group focus session.
- Create a group focus session (UC-02): Users can create a new group focus session and invite others to join.
- View participants (UC-03): Users can view the list of participants in a group focus session.
- Choose music (UC-04): The host can select background music from the app's library to play during the group focus session.
- Set a timer (UC-05): Users can set the focus timer for their private focus session or the group if they are hosts, and all participants can see the progress of the shared focus timer.
- Choose a pet (UC-06): Users can choose a virtual pet character to accompany them during focus sessions.
- Receive rewards (UC-07): Completing focus sessions can unlock rewards for the virtual pet, such as new accessories and customizations.
- Create a private focus session (UC-08): Users can start a focus session independently without group involvement.
- View Pet Inventory (UC-09): Users can view their virtual pets and accessories collection.
- Purchase a new pet (UC-10): Users can purchase new virtual pets using in-app currency or rewards.
- View Analytics (UC-11): Users can view analytics and reports on their focus habits, such as session history, average session length, and total focus time.

For more details, please refer to the corresponding sections of the System Proposal.

## 2.3. Non-functional Requirements and Design Constraints

### 1. Constraints:

- Development Resources and Time: Because of limited resources, it is necessary to prioritize core functionalities such as focus timer, virtual pets, and basic social features, with advanced features to be introduced later based on user feedback and market demands.
- Balancing Gamification and User Experience: The design must ensure that gamification elements motivate users without detracting from the core focus and learning experience.
- Device Compatibility: Ensuring compatibility across different devices requires thorough testing and development, particularly iOS and Android.
- Privacy and security: Data collection must adhere to privacy regulations.

### Non-Functional Requirements:

- Operational Requirements:
  - Compatibility with Android and iOS.
  - Interoperability with music streaming platforms and APIs if applicable.
- Performance Requirements:
  - The landing page must support 1,000 users per hour with a response time of 6 seconds or fewer.
  - The system must scale to support at least 100,000 simultaneous visits.
  - The application must have 99% uptime, excluding maintenance.

- Minimum bandwidth usage of 5mbps.
- Security Requirements:
  - Protection against unauthorized access through email and password authentication.
  - Encryption of all data.

For additional details, please refer to Section 1 of the Constraints, Non-Functional Requirements, and Feasibility Assessment.

## **2.4. System Evolution**

In Version 1 (Minimum Viable Product) MVP, we will deliver the core components of the app. These components include account authorization, creating a private or group focus room, setting a timer, and choosing a virtual pet to accompany the user's focused session. More details about each capability of Version 1 are in Section 5.3 of the System Proposal.

### **2.4.1. Version 2 Changes**

In Version 2, we will introduce several new use cases and enhancements to build upon the core functionalities delivered in the MVP. These changes will provide more advanced features to enhance user engagement and productivity.

- Integrate with a music streaming service if we can not collaborate with a music streaming service in the first version of Focus Critters: We will integrate with a music streaming service platform so that users can have a wider choice of music when creating a focus room.
- Advanced Focus Analytics: Generate and display detailed reports on user focus habits and trends.  
We will introduce Advanced Focus Analytics within the app, enabling users to generate and display detailed reports on their focus habits and trends. Users can view insights into their most productive times of day, reports on preferred focus session lengths, and analyze virtual pet companions' impact on concentration. Including advanced analytics provides users with valuable feedback, helping them optimize their focus sessions and enhancing the overall effectiveness of the app.
- Integrations with Productivity Tools and calendars:  
Users can schedule focus sessions within their calendars, integrate to-do lists for a comprehensive productivity experience, and seamlessly synchronize with third-party productivity applications. These integrations streamline the user's workflow, enhancing productivity and ensuring a cohesive focus management system.
- Challenges and achievements system:  
Users can participate in focus challenges with friends or colleagues, earn badges and achievements for completing milestones, and track their progress on leaderboards to compare with peers. This feature adds a competitive element to the app, motivating users to stay focused and achieve their goals.

### **2.4.2. Version 3 and beyond Changes**

In Version 3 and beyond, we envision additional functional capabilities and substantial enhancements to existing features. These updates will refine the app's usability, security, and engagement.

- App-blocking functionalities.  
We will implement app-blocking functionalities during focus sessions within the Focus Critters app. Users can select specific apps to block, enforce temporary restrictions on

access to distracting applications, and configure settings to personalize their app-blocking preferences. This feature helps users maintain focus by reducing potential distractions.

- Do not disturb integration.

This use case integrates the Focus Critters app with device-level Do Not Disturb modes to minimize external interruptions during focus sessions. It automatically activates Do Not Disturb mode, silencing notifications and calls to create a distraction-free environment. Users can customize Do Not Disturb behavior through user-configurable settings.

## 2.5. Document Outline

This document outline provides a comprehensive guide to the structural design of the Focus Critters application.

- Structural Model.

This section will provide a detailed breakdown of the structural model for Focus Critters, including an introduction to how the Class Diagram and Metadata descriptions are interconnected, laying the groundwork for understanding the system's structure. The Class Diagrams subsection will feature UML standard class diagrams depicting the class objects created during development and illustrating their interactions. The Metadata subsection will summarize each object within the system, highlighting its central role and providing detailed descriptions to support the class diagrams.

- Architectural Design.

The Architectural Design section will offer an in-depth overview of the overall architecture of Focus Critters, outlining the key components and their relationships. Within the Infrastructure Model subsection, we will present two deployment diagrams: the first will provide a high-level view of the system's architecture using a box-line format, and the second will offer detailed UML component and deployment diagrams showing the nodes and artifacts involved. The Hardware and Software Requirements subsection will describe the hardware for the Focus Critters application, covering both server-side and user-side components, and detail the software components, including operating systems, database management systems, and other essential software.

The Security Plan subsection will provide insights into the security precautions and measures that should be considered in the system's design to protect against various threats. This includes a comprehensive security plan chart detailing the protocols and services employed to mitigate adverse or disastrous events affecting the software system and hardware.

The User Interface Design subsection will outline the specific requirements and constraints for the user interface, present UML-designed navigation frames that explain button interactions, offer rough drafts of the mobile and web versions of Focus Critters, and detail the design of formal outputs such as reports and user notifications.

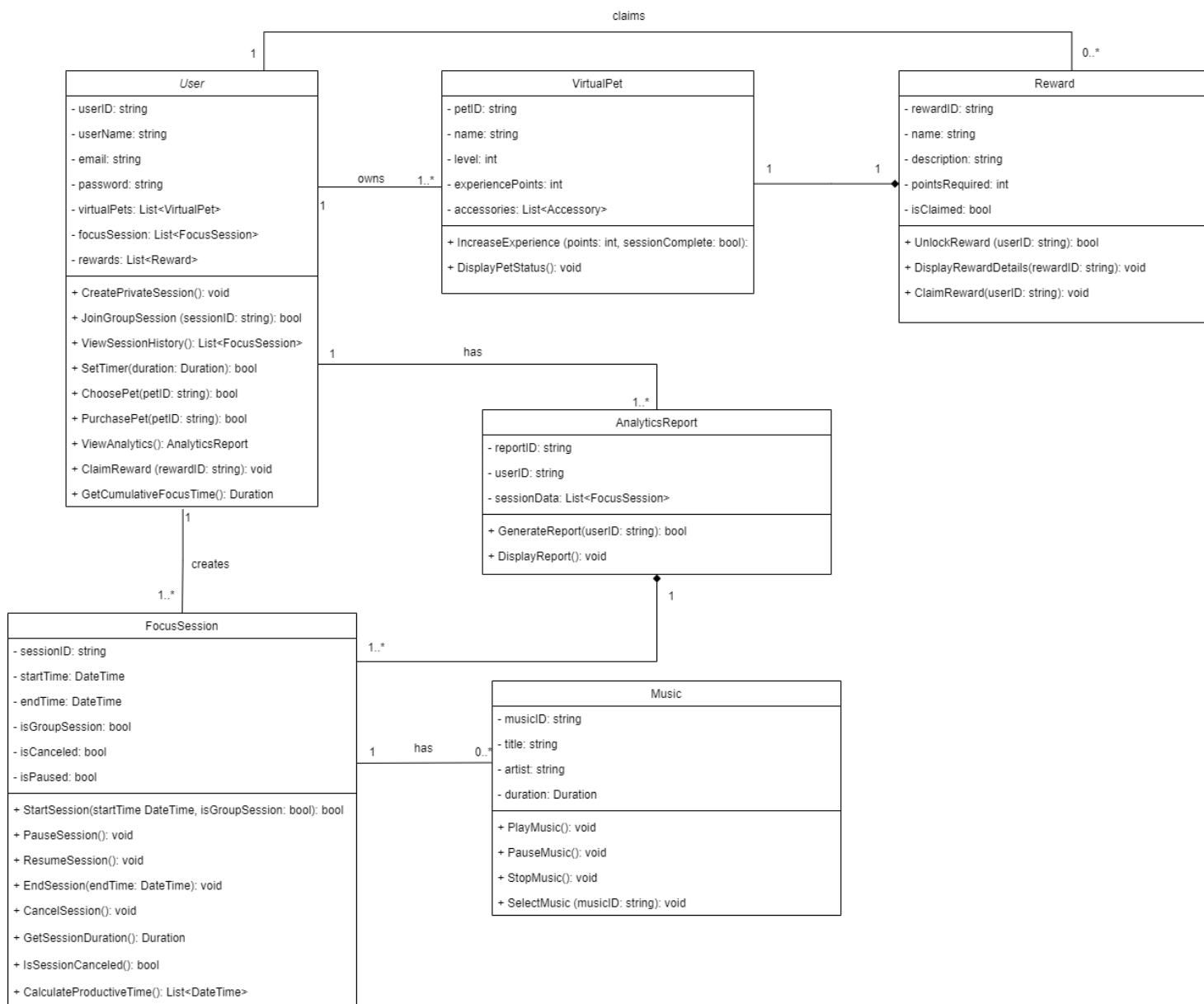
The Appendices will include a glossary of technical terminology with definitions used throughout the document, citations of sources that have contributed to the design and development of Focus Critters, and additional supporting documentation that reinforces the information presented in the main sections of the document.

### 3. Structural Model

#### 3.1. Model Introduction

This section contains an overview of the class diagrams and their metadata for the Focus Critters application. The Class Diagrams section uses the Unified Modeling Language (UML) class model to illustrate the system's structure, highlighting the class names, attributes, methods, and connections. The Metadata section provides additional information about the attribute components and operations, outlining the processing logic for the processes and detailing the behavior of each class.

#### 3.2. Class Diagrams



[Focus Critters Class Diagram drawn in Draw.io](https://draw.io)

### 3.3. Metadata

The Metadata section below provides a detailed overview of each class, including its member variables and methods, presented with pseudocode, allowing developers to understand and conceptualize the potential implementation. Below is the list of pages for each class:

1. User—Page 9
2. VirtualPet—Page 12
3. Reward—Page 14
4. FocusSession—Page 16
5. AnalyticsReport—Page 18
6. Music—Page 19



## 1. User

Description: Represents a user within the Focus Critters app.

Visibility: Public

Is Abstract: No

Additional Information:

Attributes:

Name	Description	Read Only?	Multiplicity
userID	A unique identifier is assigned to each user	Yes	1
userName	User's name	No	1
email	User's email address	No	1
Password	User's password	No	1
focusSession	List of focus sessions created by the user	No	1..*
virtualPets	List of virtual pets owned by the user	No	1..*
rewards	List of rewards earned by the user	No	1..*

Operations:

Name	Description	Is Query?	Is Polymorphic?
CreatePrivateSession	Starts a new private focus session	No	No
JoinGroupSession	Allows the user to join a group focus session using the specified session ID	No	No
SetTimer	Sets the timer duration for the user's focus session	No	No
ChoosePet	Allows the user to select a virtual pet by specifying its ID	No	No
PurchasePet	Enables the user to purchase a new virtual pet using the specified pet ID	No	No
ViewAnalytics	Generates and displays an analytics report for the user's focus habits and sessions	Yes	No
ClaimReward	Allows the user to claim a reward by providing the reward's ID	No	No

GetCumulativeFocusTime	Calculates and returns the user's cumulative focus time across all sessions	Yes	No
------------------------	-----------------------------------------------------------------------------	-----	----

### Processing Outlines

#### CreatePrivateSession():

- Initialize the session and call the method SetTimer(), SelectMusic() and StartSession ()
- Add session to the users' list of focus session

#### JoinGroupSession (sessionID: string)

- Validate sessionID
- If sessionID is valid,
  - Add the user to the session's participant list
  - Update the user's focus sessions

#### SetTimer (duration: Duration)

- Specify the desired timer duration
- Associate the timer with the active focus session
- Start the countdown

#### ChoosePet (petID: string)

- Validate petID
- If petID valid
  - Set the selected pet as active
  - Update the user's virtual pet

#### ViewAnalytic():

- Collect data from all focus sessions
- Process and analyze the data
- Generate a visual or textual report
- Display the report

#### ClaimReward (rewardID: string)

- Validate the rewardID
- If rewardID is valid
  - Check if the user meets the requirements
  - If requirements are met,

Unlock the reward (e.g., virtual pet) associated with the rewardID

Add the reward to the user's list of earned rewards

Else

Return "Requirements not met"

GetCummulativeFocusTime

Sum the duration of all completed focus sessions

Return the total focus time

## 2. VirtualPet

Description: Represents a virtual pet that accompanies users during focus sessions.

Visibility: Public

Is Abstract: No

Additional Information:

Attributes:

Name	Description	Read Only?	Multiplicity
petID	A unique identifier is assigned to each pet	Yes	1
name	The name of the virtual pet	No	1
level	The level of the virtual pet indicates its growth stage.	No	1
experiencePoints	Total experience points accumulated by the pet through user focus sessions, which contribute to leveling up the pet	No	1..*

Operations:

Name	Description	Is Query?	Is Polymorphic?
IncreaseExperience	Increases the virtual pet's experience points by the specified amount	No	No
DisplayPetStatus	Displays the current status and attributes of the virtual pet	No	No

Processing Outline:

IncreaseExperience (points: int, sessionComplete: bool):

Validate the input experience points and session completion status.

Check if the focus session was completed

If sessionCompleted is true

Add the specified experience points to the pet's current experience

Check if the pet's accumulated experience points have reached the threshold for leveling up

If a threshold is reached:

Increase the pet's level by 1

Update the pet's total experience points

Else

Update the pet's total experience points

Return "Experience points added."

Else

Return "Focus session was not completed. Experience points not added."

DisplayPetStatus ()

Retrieve the current status and attributes of the virtual pet.

Output the pet's name, level, experience points, and accessories.

Return the status information.

### 3. Reward

Description: Represents a virtual pet within the Focus Critters app.

Visibility: Public

Is Abstract: No

Additional Information:

Name	Description	Read Only?	Multiplicity
rewardID	A unique identifier is assigned to each reward	Yes	1
name	The name of the reward	No	1
description	The description of the reward	No	1
pointsRequired	Total focus time required to unlock the reward, expressed in a predefined amount of time (hours, minutes, etc.)	No	1
isClaimed	Shows if the reward has been claimed	No	1

Operations:

Name	Description	Is Query?	Is Polymorphic?
UnlockReward	Unlocks a reward for the specified user based on their achievements or accumulated points	Yes	No
DisplayRewardDetail	Displays detailed information about the selected reward, including its name, description, and requirements	No	No
ClaimReward	Allows the specified user to claim and receive the unlocked reward	No	No

Processing Outline:

UnlockReward(userID: string)

    Retrieve the user's accumulated points.

    Compare the user's accumulated points with the points required to unlock the reward

        If the accumulated points are greater than or equal to pointsRequired

            Set the reward's isClaimed attribute to false

            Return "Reward unlocked."

        Else

            Return "Not enough points to unlock the reward."

DisplayRewardDetail (rewardID: string)

Retrieve the reward details using the rewardID  
Output the reward's name, description, pointsRequired, and isClaimed status.  
Return the detailed information.

ClaimReward (userID: string)

Check if the user already claims the reward

If not claimed:

- Mark the reward as claimed by the user.
- Deliver the reward to the user (add a new virtual pet)
- Return success.

Else

- Return failure (reward already claimed)

#### 4. FocusSession

Description: Represents a focus session created by users individually or as a group host.

Visibility: Public

Is Abstract: No

Additional Information:

Attributes:

Name	Description	Read Only?	Multiplicity
sessionID	A unique identifier is assigned to each focus session	Yes	1
startTime	The start time of the focus session	No	1
endTime	The end time of the focus session	No	1
isGroupSession	Shows whether the session is a group session	No	1
isCanceled	Shows whether the session has been canceled	No	1
isPaused	Shows whether the session is currently paused	No	1

Operations:

Name	Description	Is Query?	Is Polymorphic?
StartSession	Starts a new focus session	No	No
PauseSession	Pauses the currently active focus session	No	No
ResumeSession	Resumes a paused focus session	No	No
EndSession	Concludes the focus session at the specified end time	No	No
CancelSession	Cancel the focus session, end any ongoing activities, and prevent the user from receiving rewards	No	No
GetSessionDuration	Retrieves the duration of the focus session	Yes	No
IsSessionCanceled	Check if the focus session has been canceled.	Yes	No



CalculateProductiveTime	Calculates and returns a list of productive times during the focus session	No	No
-------------------------	----------------------------------------------------------------------------	----	----

#### Processing Outline:

##### StartSession (startTime DateTime, isGroupSession: bool)

Set startTime to the provided startTime.  
Set isGroupSession to the provided isGroupSession value.  
Initialize other session attributes (isPaused, isCanceled) to false.  
Return success (true).

##### PauseSession ()

If isPaused is false:  
Set isPaused to true.  
Log the pause time.  
Return.

##### ResumeSession ()

If isPaused is true:  
Set isPaused to false.  
Log the resume time.  
Return.

##### EndSession (endTime: DateTime)

Set endTime to the provided endTime.  
Calculate the total session duration.  
If isCanceled is false and isPaused is false:  
Mark the session as completed.  
Return.

##### CancelSession ()

Set isCanceled to true.  
Terminate any ongoing activities.  
Prevent the user from receiving rewards for this session.  
Return.

##### GetSessionDuration ()

Calculate the difference between startTime and endTime.  
Return the calculated duration.

##### IsSessionCanceled ()

Check the isCanceled attribute.  
Return the cancellation status.

##### CalculateProductiveTime ()

Calculate and compile a list of productive time intervals during the session.  
Exclude any time when the session was paused.  
Return to the list of productive times.

## 5. AnalyticsReport

Description: Represents an analytics report on the user's focus habits.

Visibility: Public

Is Abstract: No

Additional Information:

Attributes:

Name	Description	Read Only?	Multiplicity
sessionID	Unique identifier assigned to each analytics report	Yes	1
userID	Identifier of the user that the report belongs to	Yes	1
sessionData	A list of data from focus sessions included in the report	No	1..*

Operations:

Name	Description	Is Query?	Is Polymorphic?
GenerateReport	Generates an analytics report for the specified user based on their focus session data	Yes	No
DisplayReport	Displays the generated analytics report, presenting insights and trends regarding the user's focus habits and productivity	No	No

Processing Outline:

GenerateReport ()

Retrieve userID to identify the user.

Collect sessionData for the user, including all relevant focus session data.

Analyze the data to identify patterns

Compile the findings into a structured report format.

Save the report with a unique sessionID.

Return the generated report.

DisplayReport ().

Return the formatted report for viewing.

## 6. Music

Description: Represents an audio source that a user can listen to during focus sessions

Visibility: Public

Is Abstract: No

Additional Information:

Attributes:

Name	Description	Read Only?	Multiplicity
sessionID	A unique identifier is assigned to each music track	Yes	1
title	The title of the music track	No	1
artist	The artist of the music track	No	1
duration	The duration of the music track	No	1

Operations:

Name	Description	Is Query?	Is Polymorphic?
PlayMusic	Starts playback of the selected music track	No	No
PauseMusic	Pauses the currently playing music track	No	No
StopMusic	Stops the playback of the currently playing music	No	No
SelectMusic	Selects a specific music track from the app's library for playback.	Yes	No

Processing Outline:

PlayMusic ()

- Retrieve the selected music track session ID.
- Load the music track from the app's library.
- Start playback of the music track.
- Monitor playback progress and duration.
- Allow user control options like pause, resume, and stop.

PauseMusic ()

- Check if there is a currently playing music track.
- Pause the playback of the music track if it is currently playing.

StopMusic ()

- Check if there is a currently playing music track.
- Stop the playback of the music track if it is currently playing.

SelectMusic (music: string)

Retrieve the user's selection of a specific music track.

Load the selected music track from the app's library.

Allow user control options like play, pause, resume, and stop for the selected track.

#### **4. Architecture Design**

##### **4.1. Architecture Overview**

This session shows the high-level architecture of the Focus Critters application. The architecture follows a client-server model, emphasizing processing distribution across servers and networked clients, mainly focusing on mobile platforms. It is structured into three tiers: client, application logic, and database. The application logic layer handles user interactions, processes data, and coordinates with external services. It will be built using Flutter 2.0. The database will be relational, and React Native will be used on the application's front end.

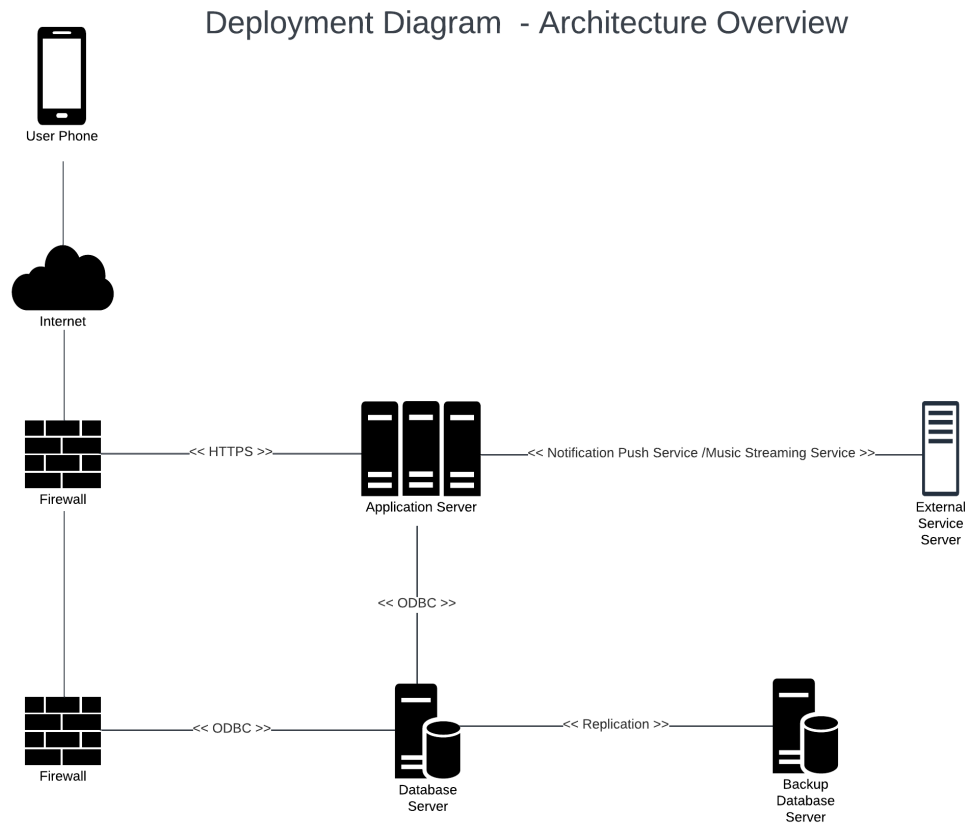
The Infrastructure Model subsection will provide detailed deployment diagrams, illustrating the system's structure and interactions between components. It will show the overall system architecture and the distribution of physical and logical entities across the infrastructure, offering insights into system scalability and connectivity.

The Hardware and Software Requirements subsection will outline the hardware components and software tools required to support the Focus Critters system. It will detail the infrastructure needed to ensure optimal performance and functionality of the application.

The Security Plan subsection highlights the security measures, considerations, and plans implemented to safeguard user data and ensure system integrity. It will cover data encryption, access control, authentication mechanisms, and threat mitigation strategies, providing a more comprehensive overview of the system's security posture.

## 4.2. Infrastructure Model

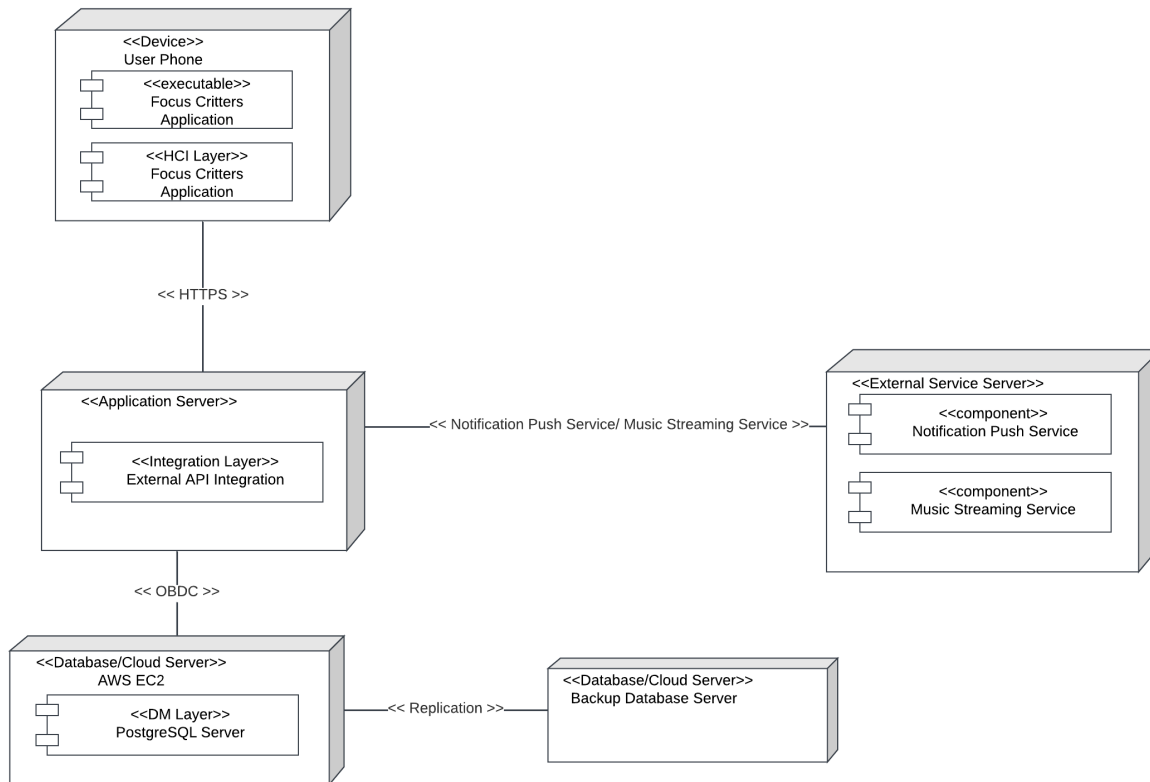
### 4.2.1. Deployment Diagram 1 – Architecture Overview



[Focus Critters Deployment Diagram - Architecture Overview on LucidChart](#)

#### 4.2.2. Deployment Diagram 2 – Nodes and Artifacts

Deployment Diagram - Nodes and Artifacts



[Focus Critters Deployment Diagram - Nodes and Artifacts on LucidChart](#)

#### 4.3. Hardware and Software Requirements

This section outlines the hardware and software components required to implement and support the Focus Critters application. It includes details on the hardware infrastructure, software dependencies, and user device requirements.

##### 4.3.1. Hardware Components

The following hardware components are required to support the Focus Critters application:

- Server-side Hardware:

1. Application Server: We will use AWS Elastic Compute Cloud (EC2) for hosting the application server to enable quick scaling of resources as needed
2. Database Server: Focus Critters use PostgreSQL hosted on AWS Relational Database Service (RDS), which supports advanced SQL features and handles concurrent data access efficiently.

3. Backup Database Server: AWS RDS Multi-AZ feature will be employed for backup database replication, providing a built-in disaster recovery solution and reducing downtime through automatic fail-over

4. Firewall and Security Appliances: Cisco ASA 5506-X will secure the network, ensuring data protection and reliable performance.

These components must be purchased new to ensure compatibility and performance for the Focus Critters application.

- User-side Hardware:

Users will need smartphones running at least Android 8.0 (Oreo) or iOS 12.0, with a minimum of 3GB RAM and 50MB free space, and a broadband internet connection for optimal performance.

#### **4.3.2. Required Software Components**

The following software components are required to implement the Focus Critters system:

1. Operating System for Servers: The application server will run on Ubuntu Server 20.04 LTS. This provides a stable, cost-effective environment for the app, ensuring reliability and support for modern software packages.
2. Database Management System: PostgreSQL 13.0 will be used for database management. PostgreSQL is known for its robustness and performance, supporting complex queries and ensuring efficient data handling.
3. Web Server Software: Nginx 1.18.0 will serve as the web server. Nginx was chosen for its high performance and scalability, making it capable of efficiently handling many concurrent connections.
4. Application Runtime: Node.js 14. x will run the application backend. Node.js was selected for its non-blocking, event-driven architecture, ideal for handling multiple simultaneous connections with high throughput.
5. Load Balancer Software: HA Proxy 2.2.3 will be used for load balancing. HA Proxy is renowned for its reliability and performance in distributing traffic across multiple servers, ensuring high availability and seamless user experience.
6. Mobile Development Framework: The app will be developed using Flutter 2.0. Flutter allows for cross-platform development, enabling the creation of high-quality apps for both Android and iOS from a single codebase.
7. Development Tools: The development team will use the Visual Studio Code and Git 2.29. Visual Studio Code is a powerful, extensible code editor, and Git is essential for version control, enabling efficient and collaborative development workflows.

### **4.4. Security Plan**

#### **4.4.1. Security Overview**

The security overview for the Focus Critters app encompasses various threats and risks that need to be considered to ensure data privacy and application integrity. These threats are categorized into different areas:

1. Disruption, Destruction, Disasters:
  - Natural Disasters: Earthquakes, floods, or storms can disrupt server operations or lead to data loss.

- Power Loss: Interruptions in the power supply can cause service downtime and data corruption.
  - Circuit Failure: Hardware failures in the server infrastructure can impact system availability.
  - Virus/Malware: Malicious software can infect devices, compromise data integrity, and disrupt app functionality.
  - DDoS (Distributed Denial of Service): Coordinated attacks can overwhelm server resources, causing service unavailability.
2. Unauthorized Access:
- Data Breach: Unauthorized access to sensitive user data can lead to legal and reputational issues.
  - Intruder: Malicious individuals attempting to gain unauthorized access to the system.
  - Insider Threats: Employees or authorized users with malicious intent can misuse their access privileges.
  - Eavesdropping: Unauthorized individuals intercept communication channels, leading to privacy breaches and data misuse.

#### 4.4.2. Security Plan

The following table presents the system architecture threats and their potential occurrences within each system component. This section will also detail the strategies and plans for mitigating each threat associated with a specific component.

Threats Components	Disruption, destruction, disasters					Unauthorized Access			
	Natural disasters	Power Loss	Circuit Failure	Virus/Malware	DDoS	Data Breach	Intruder	Insider Threats	Eavesdropping
Application Server	1	1	1	5	8		4	10, 11	
Database Server	1, 2	1, 2	1, 2	2	7, 8	3, 4	4	11	
Internet Connection									6
Backup Database	1	1	1	3	8	2	2	11	
User Devices							9	10	

1. Disaster recovery plan
2. Regularly backup and encrypt database backups
3. Regularly update and patch software to address the vulnerabilities
4. Implement robust encryption for data at rest and in transit
5. Install and maintain anti-malware and antivirus software
6. Transport Layer Security (TLS) for data transmission
7. Configure network-level protection on the database server to filter and block malicious traffic during DDoS attacks.
8. Third-party DDoS protection services
9. Encourage users to use strong passwords
10. Monitor user behavior for suspicious activities
11. Implement strict access controls and role-based permissions



## 5. User-Interface

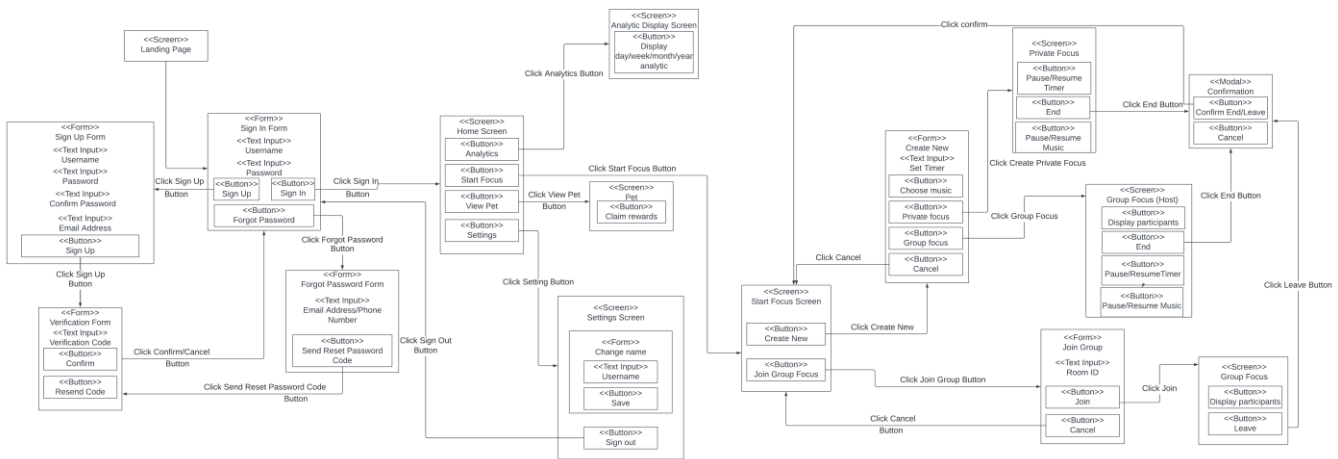
### 5.1. User-Interface Requirements and Constraints

The User Interface of the Focus Critters app will be detailed in the following sections. Our primary aim is to ensure that users can access any desired feature within the app in three clicks or fewer. The Window Navigation section will illustrate this principle by showing how each button or option intuitively leads the user through the application.

The UI Wireframes section will visually represent the user interface, including screens, layout, and navigation flow from one page to another. These wireframes will depict the minimal viable product, excluding interactions with external services like music streaming or in-app purchases, which will be integrated after partnering with relevant services.

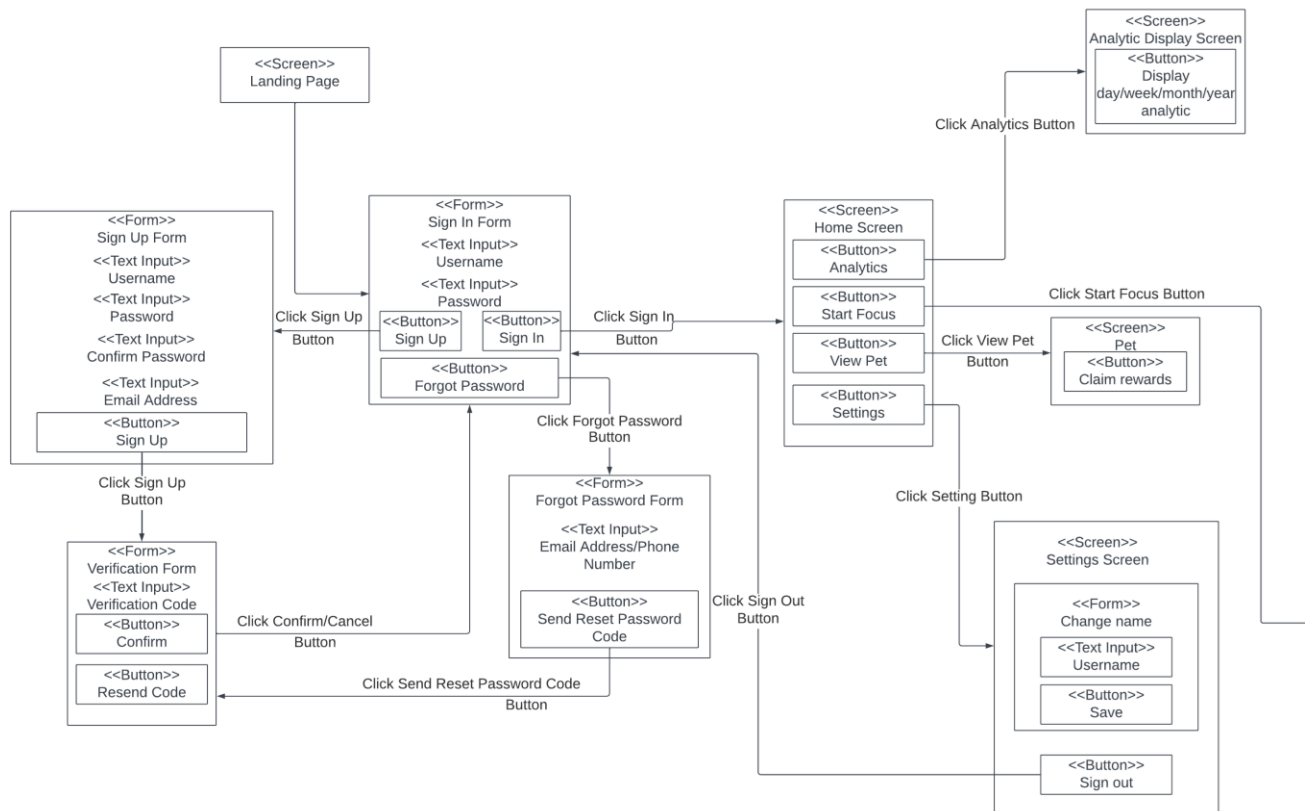
### 5.2. Window/Screen Navigation Diagram

Navigation Diagram

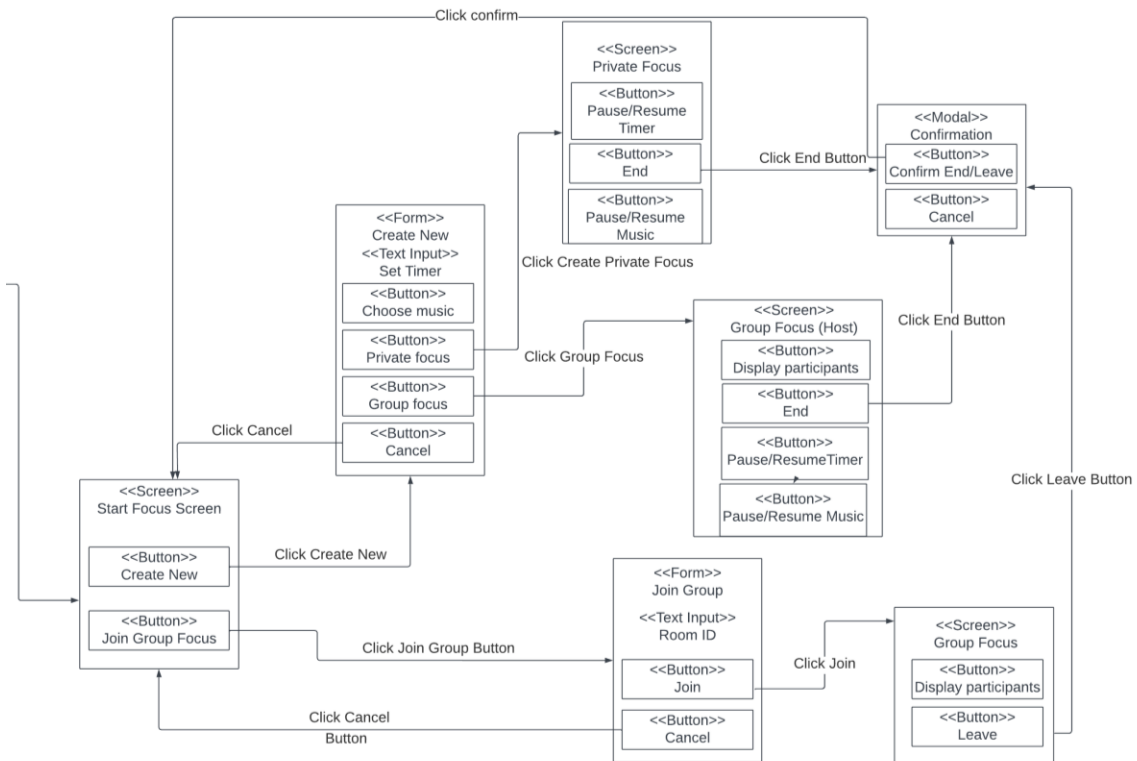


[Focus Critters Navigation Diagram on LucidChart](#)

## Left Zoom

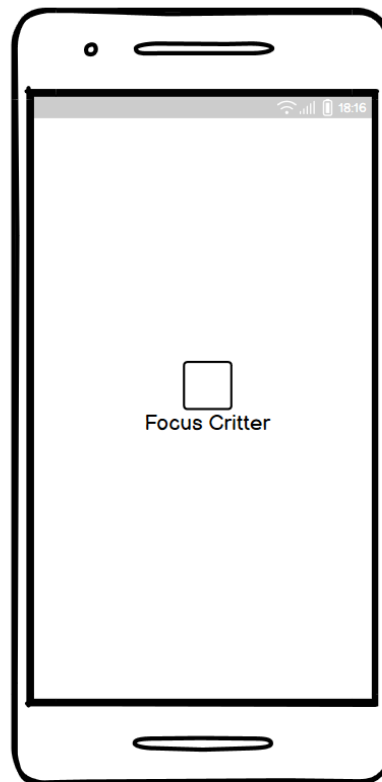


## Right Zoom



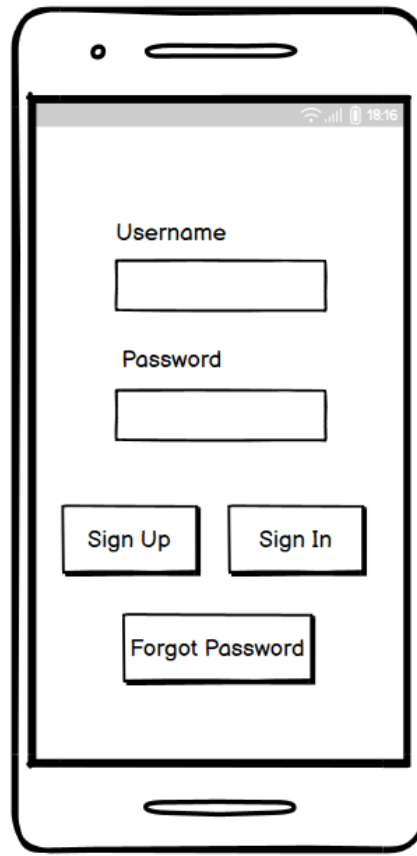
### 5.3. UI Wireframes

Below are wireframes for the Focus Critters mobile app. These frames show the desired design format, with details about specific features that should be implemented within each page. The wireframes are rough drafts that provide developers with a clear target for the design. However, it is essential to note that they only depict the minimal viable product, excluding interactions with external services like music streaming or in-app purchases, which will be integrated after partnering with relevant services.



#### Landing Page

The Landing Page is the first screen users see when they open the application. It shows the app's page icon.



### Sign-In Form

The Sign In form contains the following elements:

- Username: A text input field for entering the user's username.
- Password: A text input field for entering the user's password.

There are three main actions on this screen:

1. Sign Up button: Navigate to the Sign Up form screen where new users can register for an account.
2. Sign In button:
  - It is initially disabled when both text input fields have no values.
  - It becomes pressable once the username is entered and the password meets the registration condition (minimum length of 6 characters).
  - Navigate to the Home screen upon successful authentication.
3. Forgot Password button: Navigate to the Forgot Password form screen, where users can start the password recovery process.

< Back

Username

Password

Confirm Password

Email

Sign Up

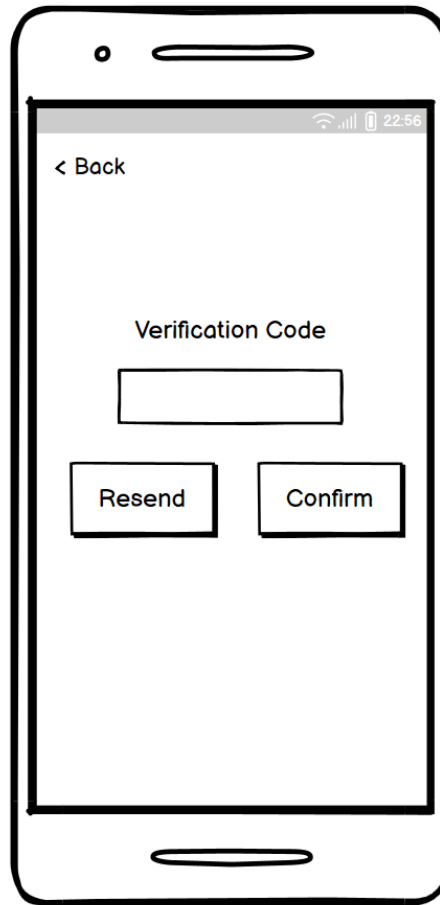
### Sign-Up Form

The Sign Up form screen contains the following elements:

- Username: A text input field for entering the desired username.
- Password: A text input field for entering the desired password.
- Confirm Password: A text input field to confirm the entered password.
- Email: A text input field for entering the user's email address.

There are two main actions on this screen:

1. Back button: Navigate back to the Sign-In screen.
2. Sign Up button: Submit the registration details.
  - The Sign-Up button should be disabled until all required fields (Username, Password, Confirm Password, Email) are filled out correctly.
  - The Password and Confirm Password fields must match to enable the button.
  - Once all conditions are met and the Sign-Up button is pressed, the form will create a new account and navigate the Verification Form.



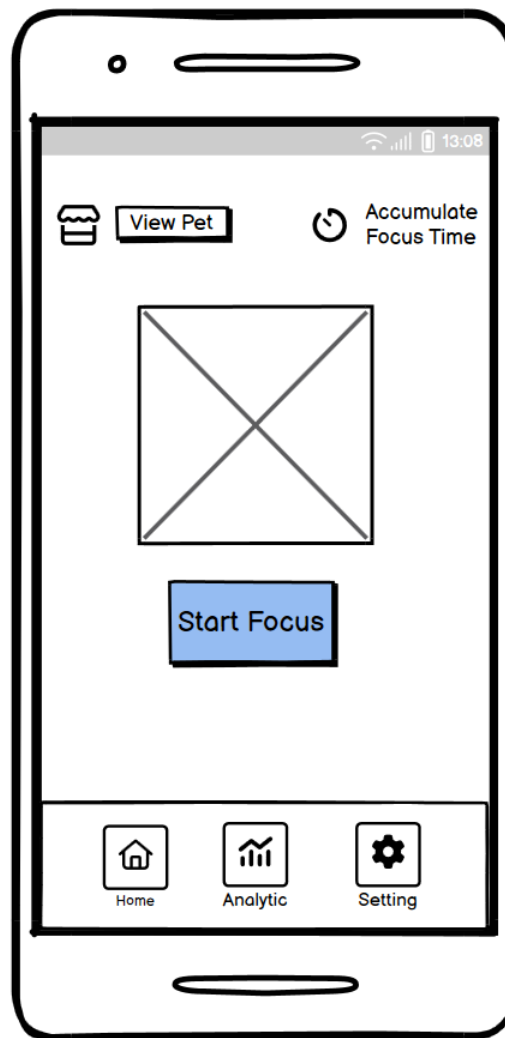
**Verification Form Screen**

The Verification Form contains the following elements:

- Verification Code: A text input field where the user enters the verification code sent to their email address.

There are three main actions on this screen:

1. Back button: Navigate back to the Sign-Up screen.
  2. Resend button: resend the verification code to the user's email if the user did not receive the code or if the code expired.
  3. Confirm button: Submit the entered verification code.
- The Confirm button should be disabled until the Verification Code field is filled out.
  - Once the correct verification code is entered and the Confirm button is pressed, the form will verify the code and navigate to the Home screen.



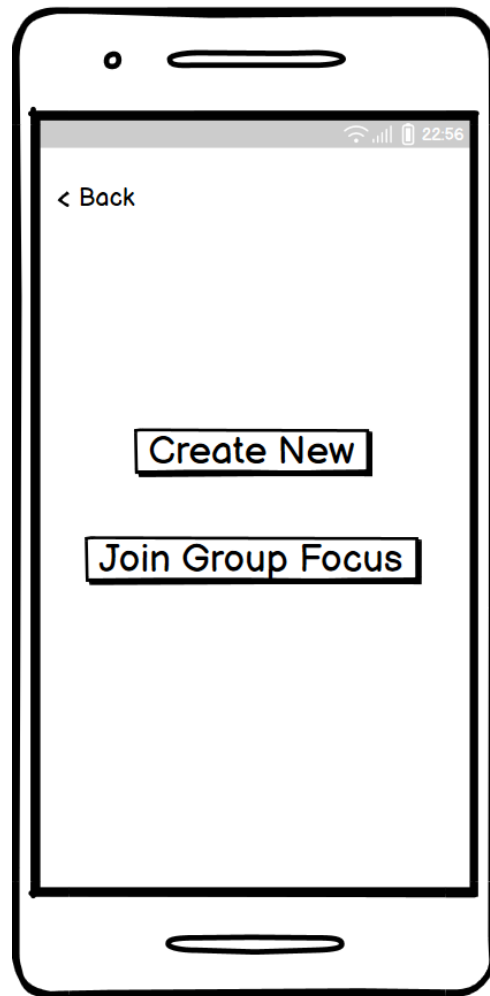
### Home Screen

The Home Screen contains the following elements:

- Accumulate Focus Time: Display the total focus time users have accumulated since they started using the app.
- Bottom navigation bar.

There are four main actions on this screen:

1. View Pet Inventory button: Navigate to View Pet Inventory Screen.
2. Start Focus button: Navigate to the Start Focus screen.
3. Analytic button: Navigate to Analytic Screen.
4. Settings button: Navigate to the Settings Screen.

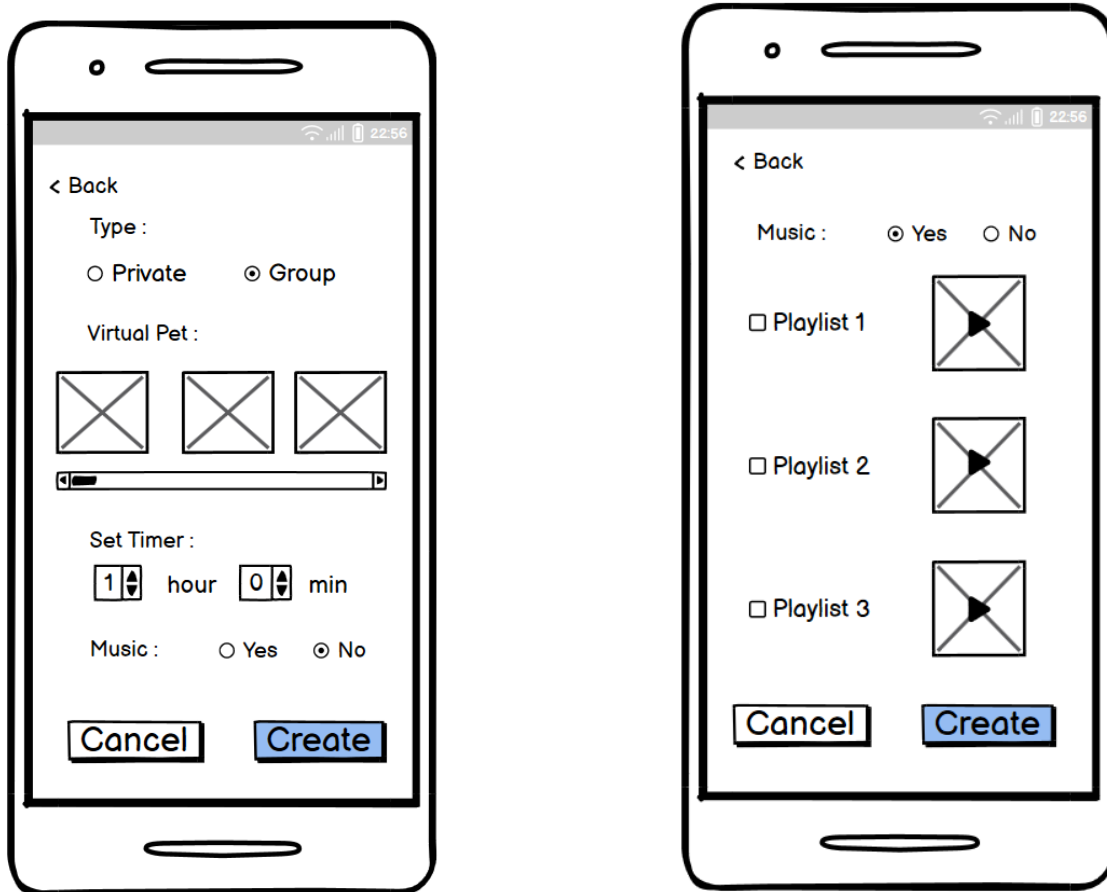


### Start Focus Screen

There are three main actions on this screen:

1. Back button: Navigate back to the Home screen.
2. Create New button: Navigate to the Create New Form.
3. Join Group button: Navigate to the Join Group Form.





### Create a New Fous Session Form

The Create New Form screen contains the following elements:

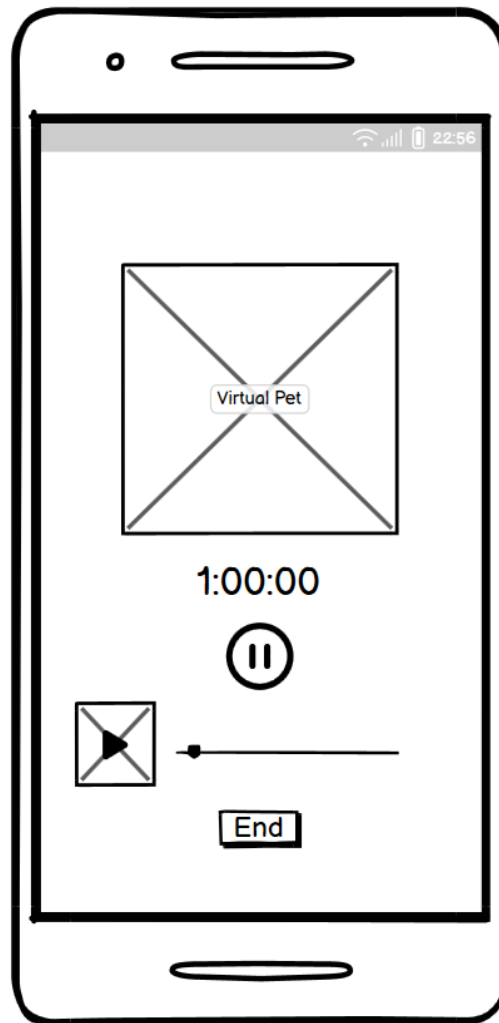
- Type: An option where the user can choose between "Private session" and "Group session." The form is the same for both options.
  - The default selection will be "Private session."
- Virtual Pet: An area with options to select a virtual pet. Users can scroll horizontally to view and choose virtual pets from their inventory.
  - Default virtual pet will be selected when users navigate to the form screen.
- Set Timer: A set of input fields to specify the duration in hours and minutes.
  - Default 15 minutes will be filled out.
- Music: A selection option to choose whether to include music ("Yes" or "No").
  - If the user chooses "Yes," The list of music playlists from the app's library will be displayed. Users can choose to select multiple lists or listen before choosing by clicking on the playlists' cover.
  - If "No," the playlist options will be hidden. The default selection will be "No."

There are three main actions on this screen:

1. Back button: Navigate back to the Start Focus screen. This is to avoid users having to scroll down to the end of the form to return to the Home screen.
2. Cancel button: Navigate back to the Start Focus screen.
3. Confirm button: Complete and save the settings.

Once the "Create" button is pressed, the form will process the settings:

- If users create a private session, navigate them to the private focus session screen.
- If users create a group session, navigate them to the group focus session screen.



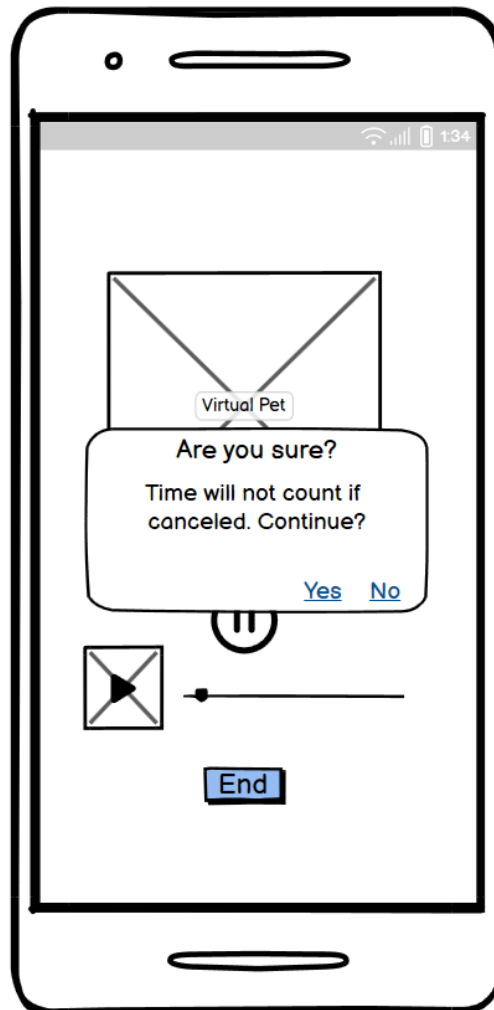
**Private Focus Session Screen**

The screen contains the following elements:

- Virtual pet: displaying selected virtual pet. If the virtual pet passes a maturity level, it should change its form but not disrupt the timer.
- Timer: A countdown timer based on the users' settings.

There are three main actions on this screen:

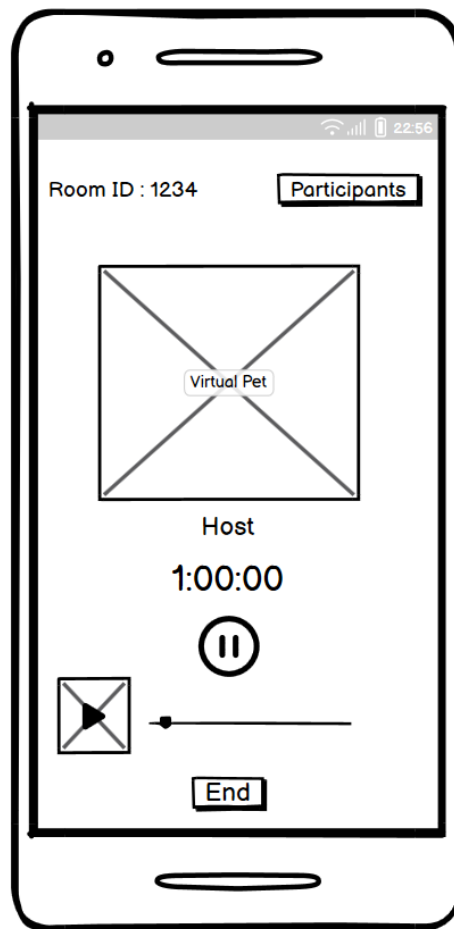
- Pause/Resume timer button: Users can choose to pause or resume the timer.
- Pause/Resume the music playlist.
- End button: Trigger a confirmation modal.



### End Confirmation Modal of Private Focus Session

When the "End" button is pressed, the timer still resumes:

- If the user selects "Yes," the session ends without saving the timer progress. Users will be navigated back to the Home Screen.
- If the user selects "No," the confirmation dialog closes.



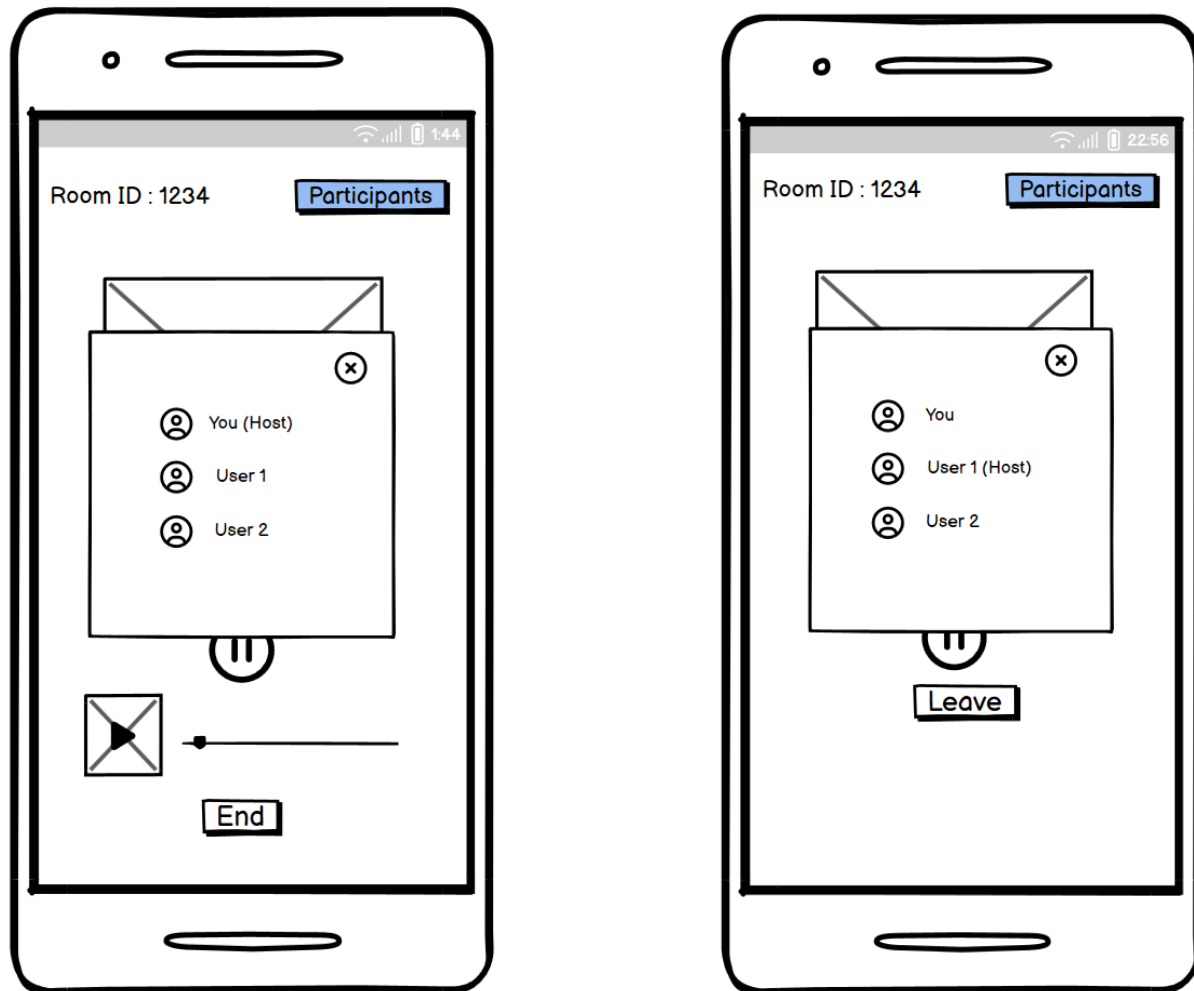
**Group Focus Session Screen (Host)**

The screen contains the following elements:

- Room ID: Displays the current room ID.
- Virtual pet: displaying selected virtual pet. The virtual pet changes form as it matures, but this does not interrupt the ongoing timer.
- Timer: A countdown timer based on the users' settings.

There are four main actions on this screen:

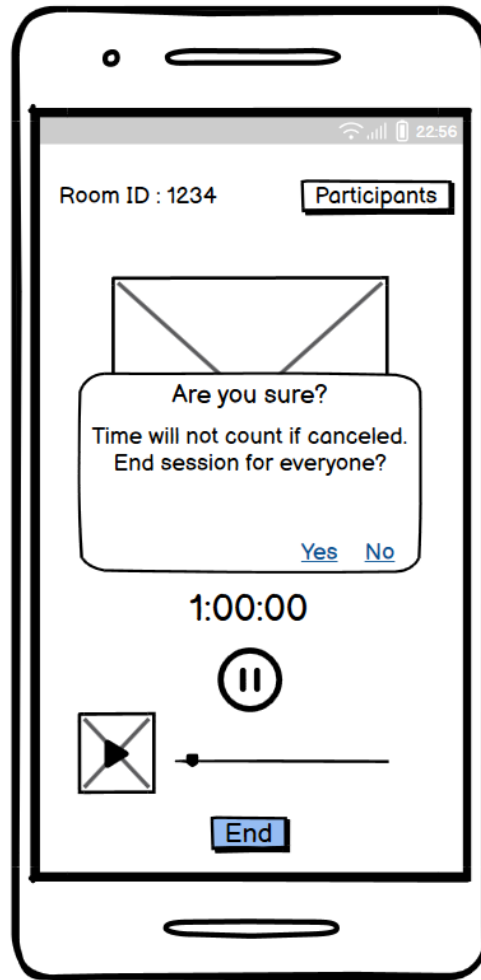
- View Participants button: trigger a modal that shows the list of participants in the room.
- Pause/Resume timer button: Users can pause or resume the timer. This action will update the timer for all participants in the room.
- Music button: Pause/Resume the music playlist. This action will update the playlist for the participants.
- End button: Trigger a confirmation modal.



### Participants Display Modal

The modal contains the following elements:

- The list of current participants, with the user display at the top.
- The closing button.



### End Confirmation Modal of Group Focus Session

When the "End" button is pressed, the timer still resumes:

If the user selects "Yes," the session ends without saving the timer progress. This also applies to all the participants in the room. Users will be navigated back to the Home Screen.

If the user selects "No," the confirmation dialog closes.

**Join Group Focus Session Form**

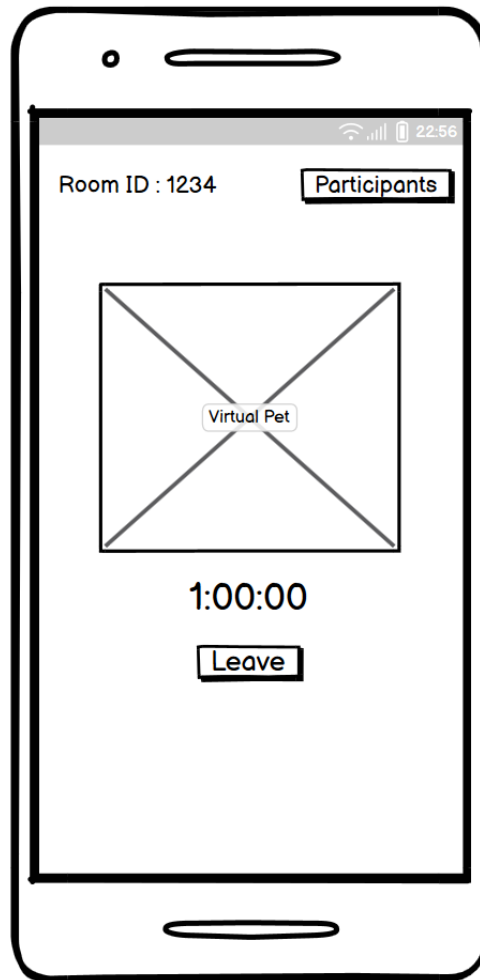
The Join Group Focus Session Form screen contains the following elements:

- Room ID: A text field where users can enter the Room ID to join a specific room.
- Virtual Pet: An area with options to select a virtual pet. Users can scroll horizontally to view and choose different virtual pets.
  - Default virtual pet will be selected when users navigate to the form screen.

There are two main actions on this screen:

1. Back/Cancel button: Navigate back to the Start Focus screen.
2. Join button: Once clicked, the app will verify if the entered Room ID is valid and exists. If the check fails, an error message is displayed.





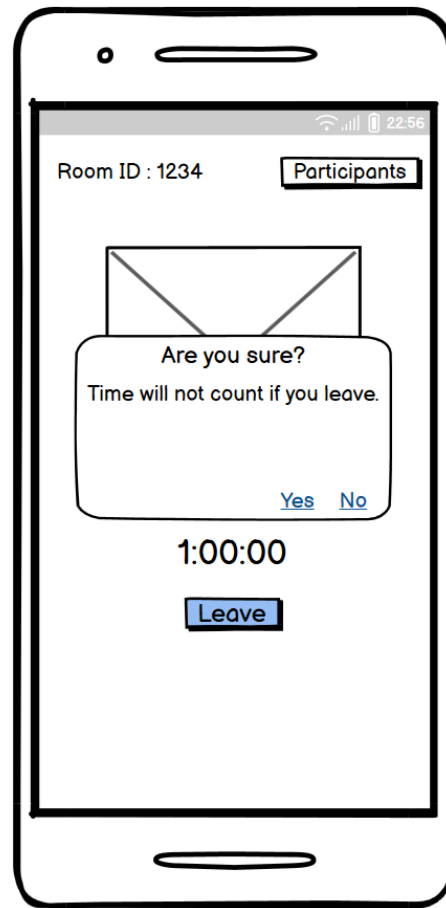
**Group Focus Session Screen (Participants)**

The screen contains the following elements:

- Room ID: shows the current room ID.
- Virtual pet: displays selected virtual pet. The virtual pet changes form as it matures, but this does not interrupt the ongoing timer.
- Timer: A countdown timer based on the host's room settings. If users join in the middle of the focus session, the time they receive will count from when they enter the room.

There are two main actions on this screen:

1. View Participants button: trigger a modal that shows the list of participants in the room.
2. Leave button: Trigger a confirmation modal.

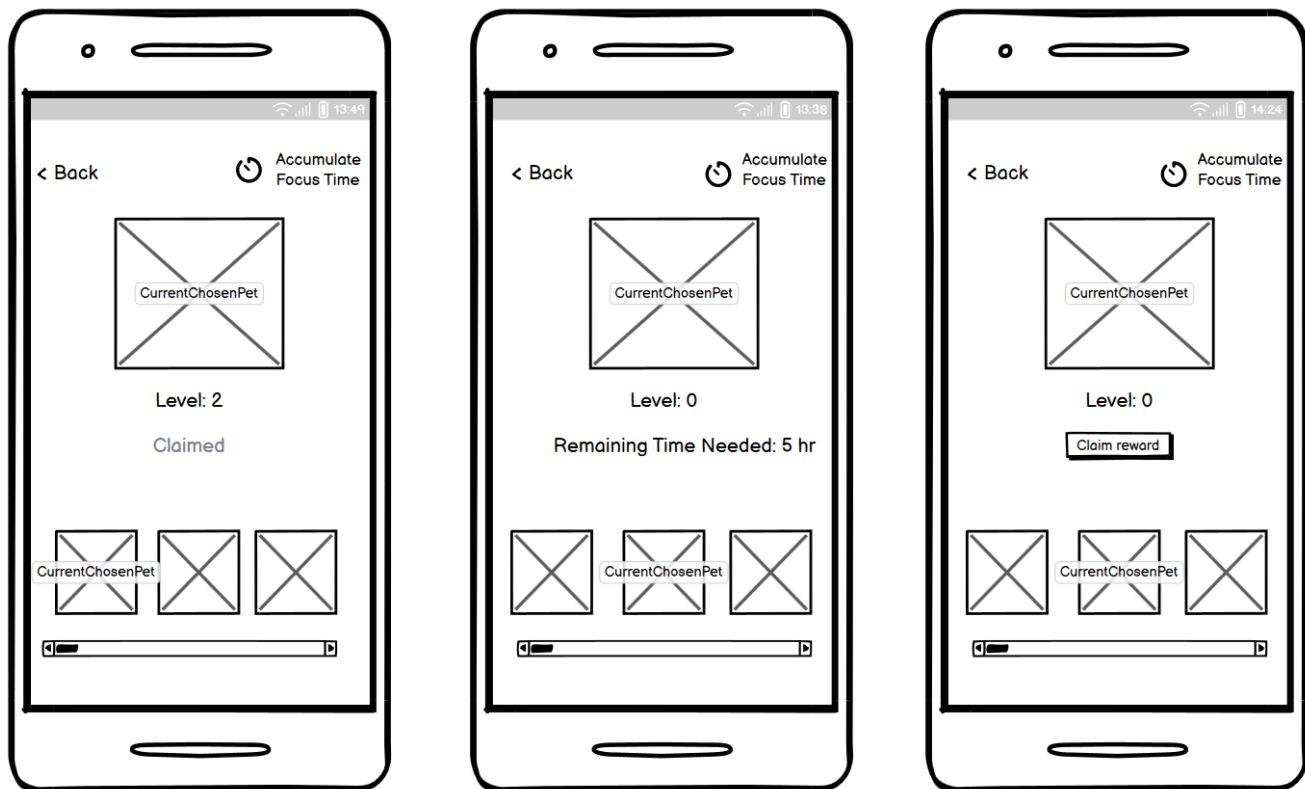


### End Confirmation Modal of Private Focus Session

When the "End" button is pressed, the timer still resumes:

- If the user selects "Yes," the session ends without saving the timer progress. Users will be navigated back to the Home Screen.

- If the user selects "No," the confirmation dialog closes.



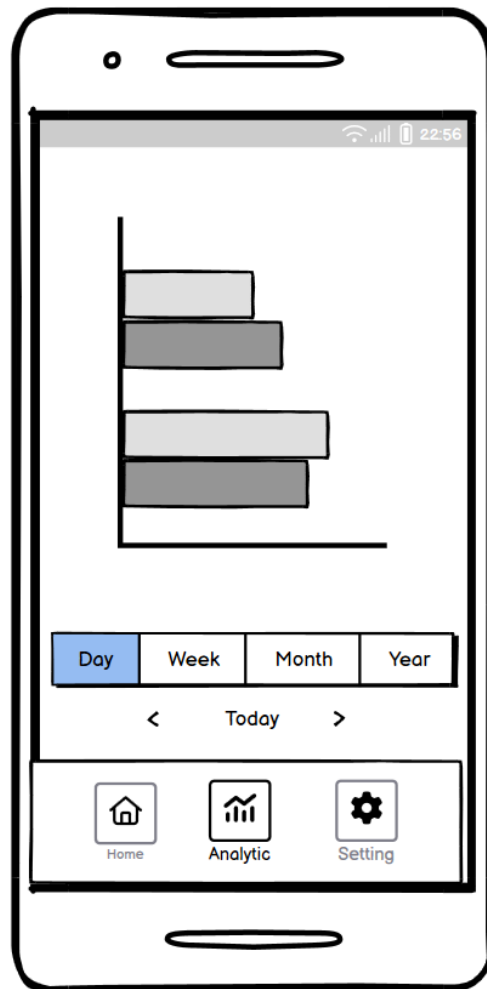
### View Pet Inventory Screen

The Join Group Focus Session Form screen contains the following elements:

- Accumulate Focus Time: Display the total focus time users have accumulated since they started using the app.
- Level of maturity of the pet.
- Reward status.
  - If the reward is not yet claimed, the screen will display the remaining time the user needs to achieve to unlock the pet.
  - The screen will display the Claim Reward button if the reward has not yet been claimed and the remaining time is 0.
  - If the reward is already claimed, the screen will display "claimed."
- Pet Selection Area: Users can tap on to display the pet they want.

There are three main actions on this screen:

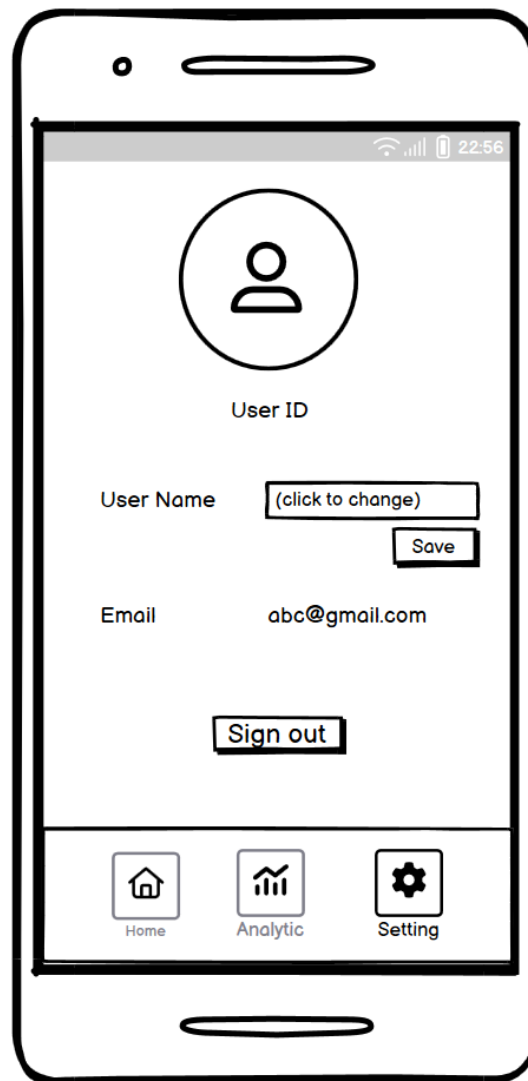
1. Back button: Navigate back to the Home screen.
2. Claim Reward button: This appears only when the pet is not claimed and the required focus time has been met. It allows users to claim the reward.
3. Pet Selection Area: Users can select different pets to view the details.



### Analytic Screen

The screen contains the following elements:

- Bar chart: Visualizes data such as total usage and average session length for different periods. Each bar represents a different metric or time segment within the selected period, such as days within a week.
- Time selection: Users can tap on the buttons labeled "Day," "Week," "Month," and "Year" to switch the view of the analytics data to the selected time.
- Date navigation: Users can navigate through different dates or periods (e.g., previous or next day, week, month, or year) to see the corresponding data.
- Bottom navigation bar:
  - Home: Take users back to the Home screen.
  - Analytic: Current screen.
  - Setting: Navigate to the Setting page.



### Setting Screen

The setting screen contains the following elements:

- User ID.
- User Name:
  - A text input field where users can click to change their username
  - This field includes a "Save" button to confirm and save any changes
- Users' Email.
- The sign-out button allows users to log out of their account and navigate to the Sign-In Form screen.
- Bottom navigation bar.
  - Home: Take users back to the Home screen
  - Analytic: Navigate to the Analytic screen
  - Setting: Current screen

#### 5.4. Reports: "Formal Output" Design

The Focus Critters app will display the main usage patterns and analytics without print-to-paper options or separate document generation.

## 6. Appendices

### 6.1. Glossary

**API:** A set of protocols and tools that allow different software applications to communicate with each other.

**AWS Elastic Compute Cloud (EC2):** A service provided by Amazon Web Services that offers scalable computing capacity for hosting applications.

**AWS Relational Database Service (RDS):** A managed relational database service by Amazon Web Services that simplifies the setup, operation, and scaling of a relational database.

**AWS RDS Multi-AZ:** A feature of AWS RDS that enhances availability and reliability by replicating databases across multiple availability zones.

**Cisco ASA 5560-X:** A firewall and security appliance that is used to secure networks, ensuring data protection and reliable performance.

**DDoS:** A cyberattack where multiple systems overwhelm a server with traffic, causing service unavailability.

**Flutter:** A mobile development framework that enables the creation of high-quality cross-platform apps from a single codebase.

**Git:** A version control system that facilitates collaborative software development and tracks changes to code.

**HA Proxy:** Load-balancing software that distributes traffic across multiple servers to ensure high availability and performance.

**Nginx:** High-performance web server software capable of handling numerous concurrent connections efficiently.

**PostgreSQL:** An advanced open-source relational database management system known for its robustness and performance.

**Ubuntu Server 20.04 LTS:** A stable, secure, and cost-effective operating system used for running servers.

**Visual Studio Code:** An extensible code editor that is used for writing and debugging software.

### 6.2. References / Bibliography

Furrmily. (n.d.). *Project example*. System Design, Seattle Pacific University

Wisdom of Paradise: The Future of Patristic Studies and Learning. (n.d.). *Project example*. System Design, Seattle Pacific University

Cameron, Andy. *Class Lecture*. System Design, Seattle Pacific University

**6.3. Supporting documentation**

N/A