**WWYD App Documentation**

**Overview:** The **WWYD** (What Would You Do) app is designed to engage users, particularly children, in interactive games aimed at promoting mental health, decision-making skills, and emotional awareness. The app offers three games that help users practice breathing exercises, memory matching, and make scenario-based choices. Each game provides a fun and educational way to encourage mindfulness, emotional regulation, and critical thinking.

**Games in the App**

**1. Breathing Activity Game**

**Objective**: To guide users through a simple breathing exercise aimed at relaxation and stress reduction.

**How it works**:

* When the user presses the "Start Breathing Exercise" button, the app instructs the user to follow a series of breathing steps:
  + **Breathe In** for 6 seconds
  + **Hold your breath** for 6 seconds
  + **Breathe Out** for 6 seconds
* These steps repeat in a cycle until the user chooses to stop the exercise.
* This game helps users learn deep breathing techniques to manage stress and anxiety.

**2. Matching Game**

**Objective**: To improve memory and cognitive skills by matching pairs of items.

**How it works**:

* The game presents a shuffled grid of items (such as fruit emojis).
* The user taps on items to reveal them, attempting to match pairs.
* If a pair is matched, it stays revealed; if not, the items are flipped back over.
* The game ends when all pairs are matched.
* This game helps with memory retention, pattern recognition, and concentration.

**3. Scenario Game**

**Objective**: To foster critical thinking and decision-making skills by presenting users with real-life scenarios and asking for their responses.

**How it works**:

* The game shows the user a scenario, such as "You see someone drop their wallet. What do you do?" with multiple response options.
* The user selects one response, and the app shows the result of that choice.
* After each decision, the next scenario appears.
* This game helps users think through real-life situations and develop healthy decision-making skills.

**Features of the App**

* **Navigation**: The app features a simple, easy-to-navigate interface with buttons for each game on the home screen.
* **Relaxation Techniques**: The Breathing Activity Game introduces users to mindfulness exercises that can help them deal with stress.
* **Memory and Learning**: The Matching Game promotes cognitive growth and memory improvement.
* **Decision-Making Practice**: The Scenario Game provides a safe space for users to practice making choices and reflect on outcomes.
* **Child-Friendly**: All games use friendly visuals and simple instructions, making them accessible for children.

**Technology Stack**

* **Android Studio**: The app is built using Android Studio and Jetpack Compose for modern UI development.
* **Navigation**: The app uses Navigation Compose to transition between different game screens smoothly.
* **Kotlin**: The app is written in Kotlin, leveraging its expressive syntax and full compatibility with Android development.

**Conclusion**

The **WWYD** app provides a mix of fun and educational games designed to engage children in positive mental health activities. By combining breathing exercises, memory games, and decision-making scenarios, the app aims to teach essential skills like relaxation, focus, and critical thinking in a playful, child-friendly format.

Breathing act.

@Composable

fun BreathingGameScreen(navController: NavHostController) {

var isBreathing by remember { mutableStateOf(false) }

var instruction by remember { mutableStateOf("Press Start to begin!") }

val breathingSteps = listOf(

"Breathe in... 🌬️" to 6000L, // 6 seconds

"Hold your breath... 🤐" to 6000L,

"Breathe out... 😮‍💨" to 6000L

)

Box(modifier = Modifier.fillMaxSize()) {

// Main Content

Column(

modifier = Modifier

.fillMaxSize()

.padding(16.dp),

verticalArrangement = Arrangement.Center,

horizontalAlignment = Alignment.CenterHorizontally

) {

if (!isBreathing) {

Button(

onClick = { isBreathing = true },

modifier = Modifier

.fillMaxWidth()

.padding(16.dp),

shape = RoundedCornerShape(8.dp)

) {

Text("Start Breathing Exercise", style = MaterialTheme.typography.bodyLarge)

}

} else {

LaunchedEffect(isBreathing) {

for ((text, duration) in breathingSteps) {

if (!isBreathing) break // Exit loop if stopped

instruction = text

delay(duration)

}

// Reset state automatically when finished

if (isBreathing) {

isBreathing = false

instruction = "Press Start to begin!"

}

}

Text(

text = instruction,

style = MaterialTheme.typography.headlineMedium,

modifier = Modifier.padding(16.dp)

)

Spacer(modifier = Modifier.height(32.dp))

Button(

onClick = {

isBreathing = false

instruction = "Press Start to begin!"

},

modifier = Modifier

.fillMaxWidth()

.padding(16.dp),

shape = RoundedCornerShape(8.dp)

) {

Text("Stop Exercise", style = MaterialTheme.typography.bodyLarge)

}

}

}

}

}

Matching

@Composable

fun MatchingGameScreen(navController: NavHostController) {

var items = remember { mutableStateListOf("🍎", "🍌", "🍎", "🍌") }

var selectedItems = remember { mutableStateOf<List<String>>(emptyList()) }

var matchedItems = remember { mutableStateListOf<String>() }

var gameOver = remember { mutableStateOf(false) }

fun resetGame() {

matchedItems.clear()

items.shuffle()

selectedItems.value = emptyList()

gameOver.value = false

}

if (gameOver.value) {

Text(text = "You matched all items! Well done!", fontSize = 20.sp)

Button(onClick = { resetGame() }) {

Text("Play Again")

}

} else {

Column(

modifier = Modifier.fillMaxSize(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

items.forEachIndexed { index, item ->

Button(

onClick = {

if (selectedItems.value.size < 2 && item !in selectedItems.value) {

selectedItems.value = selectedItems.value + item

if (selectedItems.value.size == 2) {

if (selectedItems.value[0] == selectedItems.value[1]) {

matchedItems.add(selectedItems.value[0])

if (matchedItems.size == items.size / 2) {

gameOver.value = true

}

}

selectedItems.value = emptyList()

}

}

},

modifier = Modifier

.padding(8.dp)

.size(100.dp),

shape = RoundedCornerShape(8.dp)

) {

Text(item, fontSize = 24.sp)

}

}

}

}

}

Scenario

@Composable

fun ScenarioGameScreen(navController: NavHostController) {

val scenarios = listOf(

"You see someone drop their wallet. What do you do?" to listOf("Pick it up", "Leave it", "Tell them"),

"You're at a party and feel uncomfortable. What do you do?" to listOf("Stay", "Leave", "Talk to someone")

)

var selectedScenarioIndex by remember { mutableStateOf(0) }

var scenarioOutcome by remember { mutableStateOf("") }

Column(

modifier = Modifier.fillMaxSize(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

Text(

text = scenarios[selectedScenarioIndex].first,

fontSize = 20.sp,

fontWeight = FontWeight.Bold,

modifier = Modifier.padding(16.dp)

)

scenarios[selectedScenarioIndex].second.forEachIndexed { index, choice ->

Button(

onClick = {

scenarioOutcome = "You chose: $choice"

selectedScenarioIndex = (selectedScenarioIndex + 1) % scenarios.size

},

modifier = Modifier

.padding(8.dp)

.fillMaxWidth(),

shape = RoundedCornerShape(8.dp)

) {

Text("Choice ${index + 1}: $choice")

}

}

Spacer(modifier = Modifier.height(16.dp))

if (scenarioOutcome.isNotEmpty()) {

Text(text = scenarioOutcome, fontSize = 18.sp, modifier = Modifier.padding(16.dp))

}

}

}

Home screen

@Composable

fun HomeScreen(navController: NavHostController) {

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colorScheme.background

) {

Column(

modifier = Modifier

.fillMaxSize()

.padding(16.dp),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

Text(

text = "Welcome to WWYD!",

style = MaterialTheme.typography.headlineMedium,

modifier = Modifier.padding(bottom = 24.dp)

)

GameButton(navController, "Breathing Activity Game", "breathingGame")

GameButton(navController, "Matching Game", "matchingGame")

GameButton(navController, "Scenario Game", "scenarioGame")

}

}

}

@Composable

fun GameButton(navController: NavHostController, text: String, route: String) {

Button(

onClick = { navController.navigate(route) },

modifier = Modifier

.fillMaxWidth()

.padding(vertical = 8.dp),

shape = RoundedCornerShape(8.dp)

) {

Text(text, fontSize = 18.sp, fontWeight = FontWeight.Bold)

}

}