ALGORITHM DETAILS

Algorithm Details

1. Adaptive Learning Algorithm

Purpose: Adjust difficulty of tasks based on child's performance.

Pseudo-code:

Algorithm Adaptive_Learning

Input: Task_set T, Child_profile C

Output: Updated task difficulty

1. Initialize difficulty_level ← baseline from C
2. For each task t in T do
    Present t to child
    Record performance metrics: accuracy, time_taken, errors
3. Evaluate performance_score ← f(accuracy, time_taken, errors)
4. If performance_score > Upper_threshold then
    Increase difficulty_level
  Else if performance_score < Lower_threshold then
    Decrease difficulty_level

Else

Maintain same difficulty_level

5. Store performance in database

6. Provide feedback to child (visual/audio cues)

7. Update caregiver dashboard

End

2. Gamification & Reward Algorithm

Purpose: Motivate children using points, badges, and AR effects.

Pseudo-code:

Algorithm Gamification_System

Input: Child_performance P

Output: Rewards R

1. Initialize points ← 0

2. For each completed task:

If task_success = TRUE then

points ← points + 10

Else

points ← points + 2 // encourage effort

3. If points ≥ Milestone then

Award Badge or AR_Reward

4. Update reward_history

5. Display rewards to child

End

---

3. Multimodal Interaction Algorithm

Purpose: Enable accessibility via touch, voice, or gesture.

Pseudo-code:

Algorithm Multimodal_Input

Input: User_input (touch/voice/gesture)

Output: System_action

1. Detect input_mode

2. If input_mode = voice then

Use Speech-to-Text API

Else if input_mode = gesture then

Use ARCore/ARKit to recognize gestures

Else

Process as touch input

3. Map input to action (select option, navigate, confirm)

4. Provide feedback (highlight, sound, vibration)

End

4. Progress Monitoring Algorithm

Purpose: Track improvement trends and generate caregiver insights.

Pseudo-code:

Algorithm Progress_Tracking

Input: Daily_logs D

Output: Progress_report R

1. Collect data from all sessions in D
2. Calculate weekly_metrics:
    memory_score ← avg(memory_tasks)
    attention_score ← avg(attention_tasks)
    problem_solving ← avg(problem_tasks)
3. Compare current_metrics with previous week
4. If regression detected then
    Trigger Alert to caregiver
5. Generate visual graphs
6. Send report R to caregiver/therapist dashboard

End

5. Caregiver AI Assistant Algorithm

Purpose: Summarize progress and give recommendations.

Pseudo-code:

Algorithm Caregiver_AI_Assistant

Input: Progress_data P

Output: Recommendations R

1. Analyze P for patterns (improvement, decline, stagnation)
2. If memory < threshold then
    Suggest "shorter memory sessions"
3. If attention improved then
    Highlight achievement
4. Generate plain-language summary for caregiver
5. Display suggestions in dashboard

End

6. Social Learning Algorithm

Purpose: Allow children to play cooperative cognitive games.

Pseudo-code:

Algorithm Social_Learning

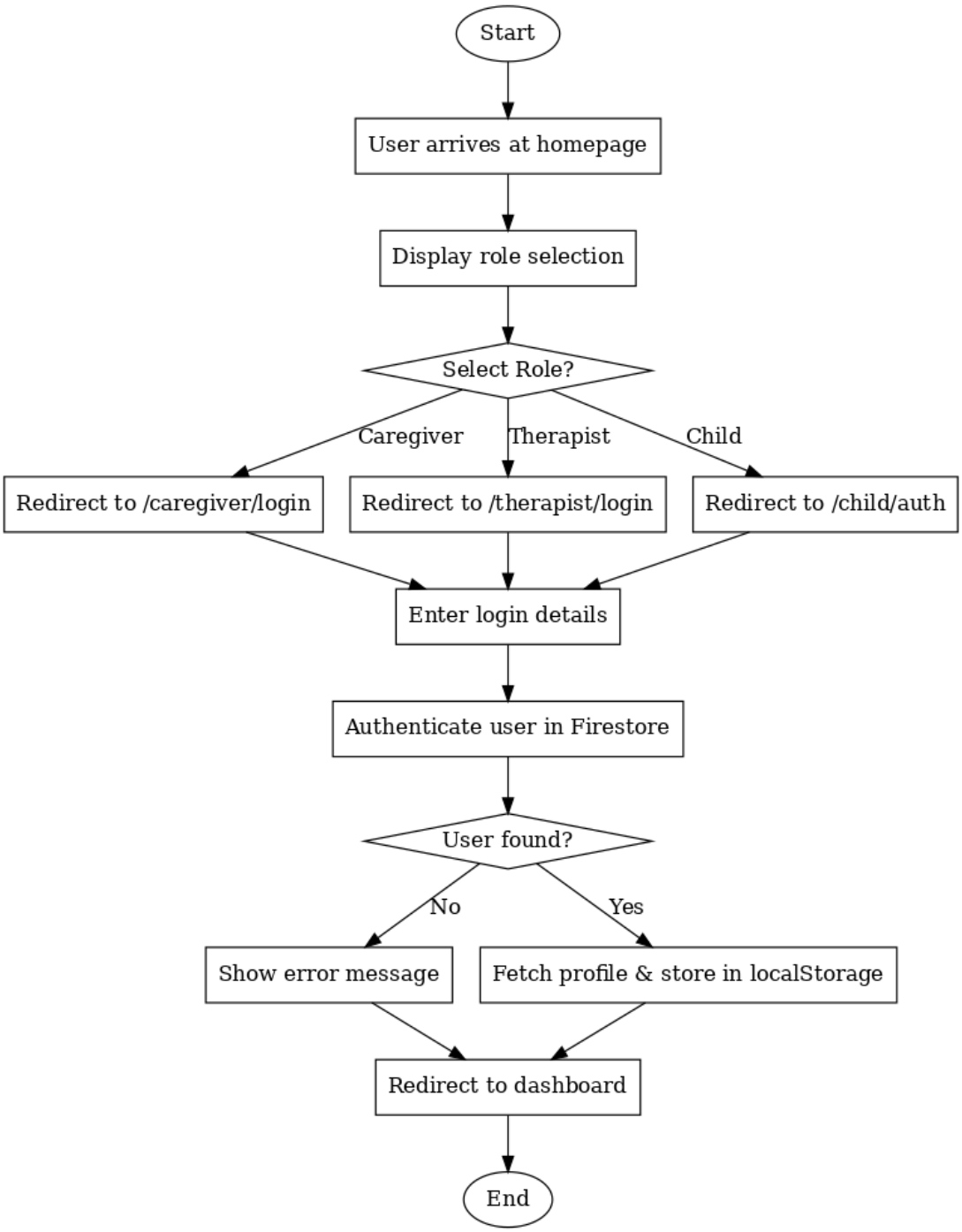
Input: Child_profile C

Output: Match and Game_session

1. Match child with peer of similar profile
2. Initiate secure WebRTC connection
3. Launch collaborative game
4. Monitor performance of both players
5. If task_success = TRUE then
    Award team points
6. Store session data in database

End

FLOW CHART :


Start
User arrives at homepage
Display role selection
Select Role?
Caregiver
Therapist
Child
Redirect to /caregiver/login
Redirect to /therapist/login
Redirect to /child/auth
Enter login details
Authenticate user in Firestore
User found?
No
Yes
Show error message
Fetch profile & store in localStorage
Redirect to dashboard
End

AUTHENTICATION ALGORITHM FLOW CHART: