# **Technical Design for POC: Alpha Wave - Phantom Reach**

## **1. System Architecture**

### **Overview**

The system comprises three main components:

1. **Real-Time Limb Tracking Module**
   * Captures residual limb movements using a standard webcam.
   * Processes the movements through a limb tracking model to create a virtual representation.
2. **AR Game Environment**
   * Developed in Unity, this module provides a gamified rehabilitation experience using the tracked limb data.
   * Includes interactive objects, levels, and feedback mechanisms.
3. **Data Management System**
   * Local and cloud storage for game data and user progress.
   * Privacy controls for data compliance.

### **Proposed Architecture**

* **Input Layer**: Standard webcam video feed.
* **Processing Layer**: Real-time tracking via a Python-based model (e.g., MediaPipe Pose or HRNet).
* **Integration Layer**: Model exported to Unity through TensorFlow or Barracuda for seamless gameplay interaction.
* **Data Layer**: Cloud services (AWS S3 or Firebase) for backup, with encrypted local storage for offline access.

## **2. Key Components and Technologies**

### **2.1. Real-Time Limb Tracking Module**

* **Input**: Video feed from a standard webcam.
* **Limb Tracking Models**:
  + **Options**: OpenPose
    - MediaPipe Pose- MediaPipe Pose can process up to 30-60 frames per second on modern hardware, making it feasible for real-time systems like webcam streams.

**Other Easy-to-Implement Alternatives:**

* + - PoseNet - Slightly slower than MediaPipe but also suitable for real-time pose detection.
    - Works well for integration into TensorFlow-based systems.
    - OpenPose - Highly accurate but computationally expensive; less suitable for low-power devices or real-time applications without GPU
    - YOLOv8 - Excellent for multitask detection (pose + object detection), but setup is more complex than MediaPipe.
  + **Criteria**: Accuracy, FPS, latency, precision, and recall.
  + **Tools**: Jupyter notebooks for model evaluation; TensorFlow for export and integration.
  + **Relevant Links**:
    - MediaPipe Pose
    - [OpenPose GitHub](https://github.com/CMU-Perceptual-Computing-Lab/openpose)
* **Integration**:
  + Preprocessing in Python.
  + TensorFlowSharp or Unity Barracuda for real-time Unity integration.

### **2.2. AR Game Environment**

* **Development Tools**: Unity (C# scripting).
* **Key Features**:
  + 3D virtual hand with real-time tracking.
  + Adjustable difficulty levels with interactive objects (e.g., cubes, balls).
  + Scoring system, progress tracking, and audio feedback.
  + Clinician-driven therapeutic objectives.
* **Relevant Links**:
  + Unity Documentation
  + [Unity TensorFlowSharp Integration](https://github.com/Unity-Technologies/ml-agents/tree/main/com.unity.ml-agents.extensions)

### **2.3. Data Management System**

* **Local Storage**:
  + JSON or SQLite for storing progress data offline.
* **Cloud Storage**:
  + Firebase or AWS S3 for backups and cross-device syncing.
* **Privacy Features**:
  + Encrypted data storage using AES.
* **Relevant Links**:
  + Firebase Documentation
  + [AWS S3 Security Best Practices](https://docs.aws.amazon.com/AmazonS3/latest/userguide/security-best-practices.html)

## **3. Workflow and Interaction**

### **3.1. User Interaction Flow**

1. User starts the game and positions themselves in front of the webcam.
2. The limb tracking module captures and processes limb movements.
3. Real-time movement data updates the virtual hand in the AR environment.
4. User interacts with game objects to complete tasks and levels.
5. Progress data is stored locally or synced to the cloud.

### **3.2. Data Processing Pipeline**

1. **Video Input**: Webcam captures video of the residual limb.
2. **Preprocessing**: Limb detection model processes input data and generates limb tracking data.
3. **Output Mapping**: Maps processed data to Unity to control the virtual hand.
4. **Feedback Loop**: Game provides real-time feedback on accuracy and performance.

## **4. Development Plan and Milestones**

### **4.1. Limb Tracking Module**

* **Tasks**:
  + Collect and annotate amputee-specific image and video data.
  + Benchmark models for real-time tracking.
  + Integrate selected model with Unity.
* **Owner**: Shoval
* **Deadline**: December 9

### **4.2. AR Game Environment**

* **Tasks**:
  + Create 3D environment and virtual hand.
  + Design levels, scoring, and feedback systems.
  + Conduct clinician interviews for therapeutic insights.
* **Owner**: Tony
* **Deadline**: December 16

### **4.3. Integration and Calibration**

* **Tasks**:
  + Link limb tracking module to Unity.
  + Implement local and cloud storage solutions.
  + Ensure privacy compliance.
* **Owner**: Vishesh
* **Deadline**: December 30

### **4.4. Usability Testing and POC Refinement**

* **Tasks**:
  + Conduct usability tests with amputees.
  + Refine mechanics and tracking based on feedback.
  + Assess clinical feasibility with therapists.
* **Owners**: Noa and Gal
* **Deadline**: January 13

## **5. Open Questions and Risks**

### **5.1. Open Questions**

1. How to efficiently collect and annotate limb-specific data for amputees?
2. Which limb tracking model balances accuracy and real-time performance best?
3. Are additional sensors (e.g., depth cameras) needed for better tracking accuracy?
4. What specific therapeutic movements should the game emphasize?
5. How can we ensure engagement for a diverse range of users?
6. What are the clinical measures of success for rehabilitation?
7. How to ensure seamless offline and online functionality?
8. Are there potential partnerships for user recruitment or clinical trials?
9. What specific regulations need to be addressed for HIPAA/GDPR compliance?
10. How to balance gaming challenge with therapeutic objectives?

### **5.2. Risks**

1. **Model Limitations**: Difficulty in achieving robust real-time tracking.
2. **User Engagement**: Limited engagement or adoption among amputees.
3. **Data Privacy**: Risks in data collection and storage compliance.
4. **Resource Constraints**: Tight deadlines and limited team resources.
5. **Feedback Misalignment**: Balancing diverse clinical and user feedback.

## **6. Conclusion**

This technical design outlines a clear roadmap for developing the POC of Alpha Wave - Phantom Reach. With innovative use of AR and gamification, the project has the potential to revolutionize rehabilitation for amputees. Timely execution, efficient resource allocation, and iterative feedback will ensure the success of the POC.