Fitness Test Web Application - Project Plan

Project Overview

A full-stack web application that uses Google MediaPipe to analyze workout form accuracy and track user fitness progress over time.

Core Features

- **10 Workout Exercises** with preset angle checkpoints
- **Real-time Form Analysis** using MediaPipe pose detection
- **Timer-based Sessions** with rep counting
- **User Authentication** and progress tracking
- **Report Generation** with progression analytics
- **Responsive Web Interface**

Technology Stack

Frontend

- **React.js** - Main UI framework
- **MediaPipe** - Pose detection and analysis
- **Chart.js/Recharts** - Progress visualization
- **Tailwind CSS** - Styling
- **WebRTC** - Camera access

Backend

- **Node.js + Express** - API server
- **MongoDB/PostgreSQL** - Database
- **JWT** - Authentication
- **Socket.io** - Real-time updates (optional)

MediaPipe Integration

- **@mediapipe/pose** - Pose landmark detection
- **@mediapipe/camera_utils** - Camera utilities
- **@mediapipe/drawing_utils** - Visualization

Database Schema

Users Collection/Table

```javascript
{
  _id: ObjectId,
  username: String,
  email: String,
  password: String (hashed),
  profile: {
    age: Number,
    height: Number,
    weight: Number,
    fitnessLevel: String
  },
  createdAt: Date,
  updatedAt: Date
}
```

Exercises Collection/Table

```javascript
{
  _id: ObjectId,
  name: String,
  description: String,
  targetMuscles: [String],
  checkpoints: [
    {
      keypoint1: String, // e.g., "left_shoulder"
      keypoint2: String, // e.g., "left_elbow"
      keypoint3: String, // e.g., "left_wrist"
      targetAngle: Number,
      tolerance: Number,
      phase: String // "up" or "down"
    }
  ],
  instructions: [String],
  difficulty: String
}
```

Workout Sessions Collection/Table

```javascript
{
  _id: ObjectId,
  userId: ObjectId,
  exerciseId: ObjectId,
  sessionDate: Date,
  duration: Number, // seconds
  totalReps: Number,
  correctReps: Number,
  accuracy: Number, // percentage
  repDetails: [
    {
      repNumber: Number,
      angles: [Number], // actual angles measured
      accuracy: Number,
      timestamp: Number
    }
  ]
}
```

Ten Workout Exercises & Angle Checkpoints

1. Push-ups

- **Key Points**: Shoulder-Elbow-Wrist angle
- **Target Angles**:
  - Down: ~90° (elbow angle)
  - Up: ~170° (elbow angle)

2. Squats

- **Key Points**: Hip-Knee-Ankle angle
- **Target Angles**:
  - Down: ~90° (knee angle)
  - Up: ~170° (knee angle)

3. Lunges

- **Key Points**: Hip-Knee-Ankle (front leg)
- **Target Angles**:
  - Down: ~90° (front knee)

- Up: ~170° (front knee)

4. Jumping Jacks

- **Key Points**: Shoulder-Hip-Knee alignment
- **Target Angles**:
  - Open: Arms ~160°, Legs ~45°
  - Closed: Arms ~20°, Legs ~180°

5. Burpees

- **Key Points**: Multiple phase tracking
- **Target Angles**: Varies by phase

6. Mountain Climbers

- **Key Points**: Hip-Knee-Ankle (active leg)
- **Target Angles**: Alternating leg positions

7. Plank Hold

- **Key Points**: Shoulder-Hip-Ankle alignment
- **Target Angles**: ~180° body line

8. High Knees

- **Key Points**: Hip-Knee angle
- **Target Angles**: Knee lift to ~90°

9. Tricep Dips

- **Key Points**: Shoulder-Elbow-Wrist
- **Target Angles**:
  - Down: ~90° (elbow)
  - Up: ~170° (elbow)

10. Sit-ups

- **Key Points**: Hip-Shoulder alignment
- **Target Angles**:
  - Down: ~180° (torso-thigh)
  - Up: ~45° (torso-thigh)

Development Phases

Phase 1: Core Setup (Week 1-2)

- [ ] Set up development environment
- [ ] Create basic React app structure
- [ ] Implement user authentication
- [ ] Set up database and basic schemas
- [ ] Integrate MediaPipe pose detection

Phase 2: Exercise Engine (Week 3-4)

- [ ] Implement angle calculation algorithms
- [ ] Create exercise configuration system
- [ ] Build rep counting logic
- [ ] Add timer functionality
- [ ] Create exercise selection UI

Phase 3: Accuracy System (Week 5-6)

- [ ] Develop form analysis algorithms

- [ ] Implement real-time feedback
- [ ] Create accuracy scoring system
- [ ] Add visual pose overlay
- [ ] Build calibration system

Phase 4: User Experience (Week 7-8)

- [ ] Create workout session flow
- [ ] Implement progress tracking
- [ ] Build report generation
- [ ] Add data visualization
- [ ] Create responsive design

Phase 5: Testing & Polish (Week 9-10)

- [ ] Performance optimization
- [ ] Cross-browser testing
- [ ] User testing and feedback
- [ ] Bug fixes and improvements
- [ ] Deployment preparation

Key Algorithms

Angle Calculation

```javascript
function calculateAngle(point1, point2, point3) {
  // Calculate vectors
  const vector1 = [point1.x - point2.x, point1.y - point2.y];
  const vector2 = [point3.x - point2.x, point3.y - point2.y];

  // Calculate angle using dot product
  const dotProduct = vector1[0] * vector2[0] + vector1[1] * vector2[1];
  const magnitude1 = Math.sqrt(vector1[0] ** 2 + vector1[1] ** 2);
  const magnitude2 = Math.sqrt(vector2[0] ** 2 + vector2[1] ** 2);

  const angle = Math.acos(dotProduct / (magnitude1 * magnitude2));
  return angle * (180 / Math.PI); // Convert to degrees
}
```

Rep Detection Logic

```javascript
function detectRep(currentAngle, targetAngle, tolerance, phase) {
  const isInRange = Math.abs(currentAngle - targetAngle) <= tolerance;

  if (phase === 'down' && isInRange && !this.inDownPhase) {
    this.inDownPhase = true;
    this.inUpPhase = false;
  } else if (phase === 'up' && isInRange && this.inDownPhase && !this.inUpPhase) {
    this.inUpPhase = true;
    this.repCount++;
    this.inDownPhase = false;
  }

  return this.repCount;
}
```

API Endpoints

Authentication

- `POST /api/auth/register` - User registration
- `POST /api/auth/login` - User login

- `POST /api/auth/logout` - User logout
- `GET /api/auth/profile` - Get user profile

Exercises

- `GET /api/exercises` - Get all exercises
- `GET /api/exercises/:id` - Get specific exercise
- `POST /api/exercises` - Create exercise (admin)

Workout Sessions

- `POST /api/sessions` - Start workout session
- `PUT /api/sessions/:id` - Update session progress
- `GET /api/sessions/user/:userId` - Get user sessions
- `GET /api/sessions/:id` - Get specific session

Reports

- `GET /api/reports/progress/:userId` - Get progress report
- `GET /api/reports/session/:sessionId` - Get session report

Deployment Considerations

- **Frontend**: Vercel, Netlify, or AWS S3
- **Backend**: Railway, Heroku, or AWS EC2
- **Database**: MongoDB Atlas or AWS RDS
- **Media**: CloudFront CDN for static assets

Security & Privacy

- Secure JWT implementation
- Video data processing locally (no upload)
- HTTPS enforcement
- Rate limiting on API endpoints
- Input validation and sanitization