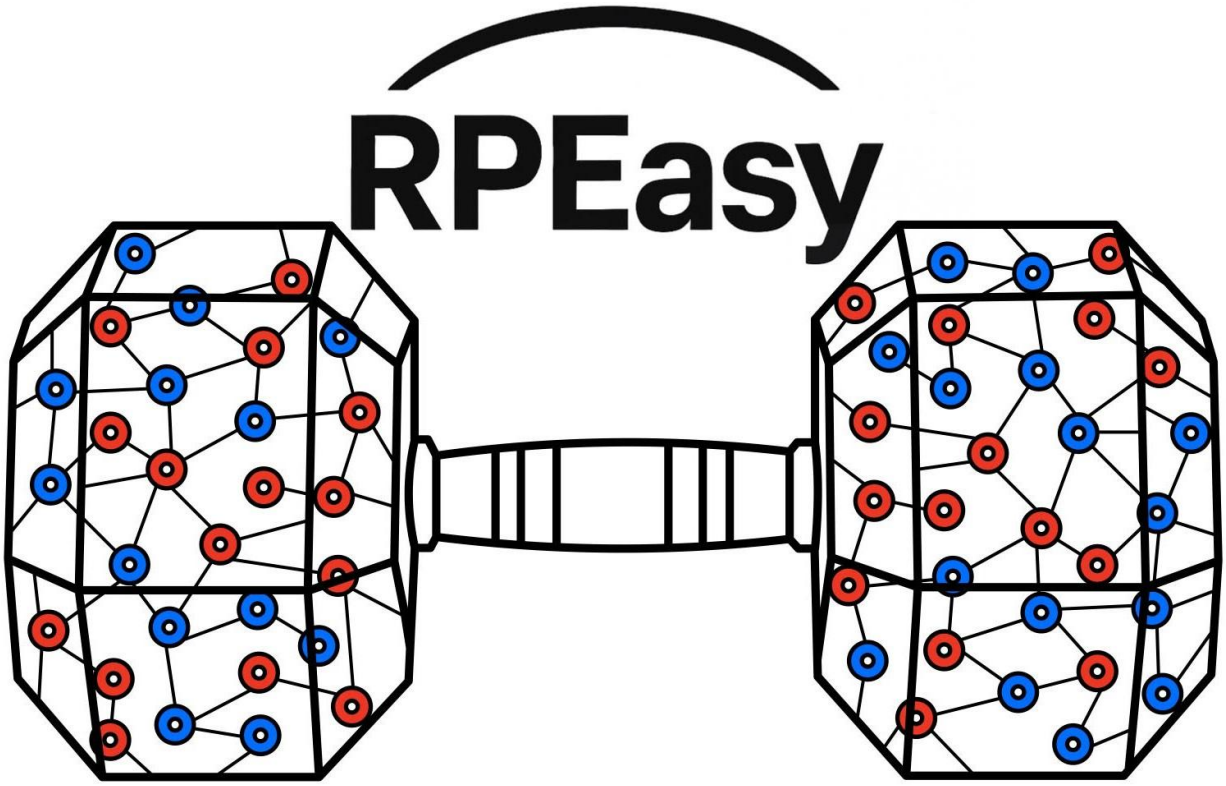
ELEC 490/498 Project Blueprint



Project title: RPEasy

Group #: 53

Group members: 1) Luke Hunker (20lrh5) (20287422)
2) Avedis Boudakian (20ajb27) (20287296)
3) Ajmal Niazi (21an18) (20300697)
4) Rhys Blakeney (20wrb1) (20283269)

Email to faculty supervisor(s): Karen Rudie

Submission date: 11/03/2025

Executive Summary

Our project is an AI-based tool that can predict an athlete's RPE on a scale of 1-10 using physical attributes such as bar speed, facial strain, and form breakdown, taken from a video input.The minimum viable product (MVP) is a functional desktop program that accepts a squat, bench press, or deadlift video and returns an RPE score based on bar tracking, pose estimation, and facial strain data. The device must possess prediction reliability of ±0.5 using the 1–10 RPE scale and maintain detection reliability, with minimal response latency.

The key design specifications for this project include high model accuracy, reliability, and responsiveness in RPE prediction. The system must take a 1080p, 60 fps RGB video input with a stable frame rate tolerance of ±2 frames. The performance requirements include an RPE prediction accuracy of ±0.5 on a 1–10 scale using an LGBM Regressor, supported by ≥95% accuracy in pose detection (MMPose), facial strain detection (OpenFace), and bar speed tracking (MMDetection). The user interface must display the resulting RPE value as text or a visual output with a latency of ≤0.5s. The optional enhanced specifications include improving the RPE accuracy to ±0.25 and increasing robustness under suboptimal recording conditions.

Our design strategy relies on design iteration to improve accuracy metrics during back-end development, taking advantage of the modular nature of the overall design to enable different team members to work on different problems concurrently. We break up our development cycle with weekly status updates to our supervisor giving information on work completed and next steps. A systematic validation plan will first verify the accuracy and reliability of each computer vision module (OpenFace, MMDetection, and MMPose). The system's overall performance will then be validated by testing the final LGBM regressor model against the ±0.5 RPE prediction accuracy target. The final application will be a novel tool for evaluating workout intensity, fulfilling a useful niche that is currently unoccupied.

Table of Contents

1. Introduction

The Rate of Perceived Exertion (RPE) is a subjective rating scale that reflects how physically demanding a set feels to a lifter. RPE is a crucial metric when weightlifting, as it helps determine an athlete's training intensity, monitor fatigue, and guide progression. Two common problems that all weightlifting athletes come across during their training are undertraining and overtraining [1][2]. A low RPE reveals that an athlete is undertraining and could have performed additional repetitions, creating a higher muscle growth stimulus. On the other end, a high RPE can be advantageous to determine if an athlete is overtraining.

Developing an AI-based tool that can predict an athlete's RPE (using physical attributes such as bar speed, facial strain, and form breakdown, taken from video) can help athletes determine if they are undertraining or overtraining. Currently, there is no product on the market that can help athletes determine their training effectiveness and reduce fatigue by assessing their RPE. This tool would be useful for all levels of weightlifting athletes, as an unbiased perspective on their RPE can provide important training insights.

This document outlines the complete design and implementation plan for the development of RPEasy, an AI-based tool that predicts an athlete's Rate of Perceived Exertion (RPE) using computer vision and machine learning. It contains a thorough explanation of the system specifications, design issues, project goals, and techniques used to produce an accurate RPE prediction. To extract physical attributes from a video input as a quantitative measure, the system combines three essential modules: pose detection, facial strain analysis, and bar speed tracking. An LGBM Regressor model is then used to process these features and produce an RPE score on a scale of 1 to 10.

The tools, design process, and validation techniques used to meet the system's accuracy requirements and dependability are covered in the remaining portion of the document. Methodology and work breakdown structure, validation and testing processes for each computer vision module, risk mitigation techniques, milestones, budgetary allocation, ethical and environmental considerations, and progress to date are all covered.

1.1 Design Problem

The core technical challenge involves the design and development of a reliable Artificial Intelligence (AI) tool that uses computer vision to accurately predict a weightlifter's Rate of Perceived Exertion (RPE) from provided video input. RPE can act as a crucial metric for measuring training intensity and progression, however an athlete's self-assessment of RPE is frequently prone to errors, leading to patterns of overtraining or undertraining [1][2].

The primary task of the AI tool is to create an efficient and smooth system that takes video footage of key weight-lifting exercises (squat, bench press, deadlift), automatically processes the input to produce a final RPE prediction value for the athlete to use to optimize their training regimen. This system will utilize multiple computer vision algorithms to quantify physical attributes or indicators of exertion, specifically tracking and measuring key elements of the video feed including bar speed, body posture, and facial strain. The tool will then use the extracted parameters as inputs to a machine learning model that will output the final RPE score.

The primary constraint will be ensuring the accuracy and robustness of the tool. The product must provide a data-driven, consistent, and unbiased perspective on the difficulty of the athlete's set. The insight provided by this tool on RPE will be beneficial for beginner and intermediate lifters and its successful development will fill a current gap in the market, providing an AI-backed objective aid to help athletes optimize their training effectiveness and prevent injury.

1.2. System specifications

Table 1: Required system requirements

Functional requirements	**Target value**	**Tolerance**
System Goal	Predict athlete's Rate of Perceived Exertion (RPE)	None
Core Lifts are supported	Squat, Bench press, Deadlift	None
Video Feature Extraction	Extract Components from video: bar speed, facial strain, body posture	None
Interface requirements		
Camera Input Resolution	1080p	None

Camera Input Frame Rate	60 fps	Expected stable frame rate (±2 frames)
Model Input Data Type	RGB video stream	None
Final RPE output	RPE digit (float rounded to integer for display)	None
Performance requirements		
RPE Prediction Accuracy (LGBM Regressor)	1-10 RPE scale	±0.5
Pose Detection Accuracy (MMPose)	≥95% accurate detection of body posture and joint positions	None
Facial Strain Detection Accuracy (Openface)	≥95% accurate face detection and strain analysis	None
Change in Bar Speed (MMDetection)	≥95% accurate bar speed detection (in seconds)	None
User Interface	Text output or visual output	Display latency ≤0.5s

Table 2: Optional Enhanced System Requirements

Functional requirements	**Target value**	**Tolerance**
RPE logging and visualization	RPE results visualized over multiple training sessions	
Interface requirements		
Cloud/Remote Access	Model runs on cloud server/API	
Mobile Application Interface	Application (IOS/Android) for video upload, user input, real-time processing	
Performance requirements		
RPE Prediction Accuracy	±0.25 for a 1-10 RPE Scale	None
Robustness	Provide accurate predictions under suboptimal recording conditions	None

The overall system goal should be an AI-based tool that is able to predict and output an RPE value using machine learning and computer vision. The system should be able to support the 3 core lifts of bench press, squat, and deadlift. This will be done by extracting video features such as bar speed, facial strain, and body posture.

For interface requirements, the input video should be 1080p, 60 fps (RGB video stream) with an expected stable frame rate (±2 frames) to ensure consistency in the dataset. This ensures that minor lighting variations or sensor noise do not significantly affect feature

extraction. The interface will display the final RPE output to the front end from the value determined from the backend calculations.

The performance requirements for this project require the model to calculate and predict an RPE value of 1-10 using the LGBM Regressor model with a tolerance of ±0.5, which was chosen for the scope of this project. The LGBM Regressor will take 3 inputs: pose detection, facial strain detection, and change in bar speed. The pose detection, facial strain detection, and change in bar speed detection accuracy should be ≥95% with no tolerance. These values include OpenFace facial action units (float), MMPose pixel x-y coordinates, and MMDetection pixel x-y coordinates. The float value, along with the change in x-y coordinates, should output with ≥95% accuracy.

These values were chosen as the system-level acceptance criteria needs to meet the ±0.5 RPE tolerance budget. Having room for only a 5% error in each input to the model allows the predicted result to meet the overall tolerance budget. The user interface should display the resulting RPE with a text output or visual output that has a display latency tolerance of ≤0.5s, which is negligible.

The optional system specifications include adding a functional requirement of logging RPE values for the user and visualizing them on a graph. This way the user can acquire a clearer picture of the intensity of their overall training. The interface requirements include creating a mobile application interface on IOS/Android for the project. It also includes running the model on a cloud server/API instead of the user's phone to speed up calculations. The performance requirements would be an RPE prediction accuracy of ±0.25 for a 1-10 RPE scale, which is an improvement on the MVP prediction accuracy of ±0.5. The final addition is to increase model robustness by training it on videos with suboptimal recording conditions.

2. Methodology/Work Breakdown Structure

2.1 Approach

The different stages of this project require different strategies during different phases of development. The differences in strategy largely relate to how labour is divided and what resources are needed. Throughout our design process we will be using one- and two-week sprint cycles. Every week we record our progress made during the previous week and plan the progress we intend to make during the next week. Every two weeks we will be meeting with our supervisor and her graduate student for progress updates and feedback. Together, these one- and two-week cycles will govern our short-term progress week-to-week.

During the work for the first two milestones, data collection and computer vision model setup, smaller modular components of the computer vision system (each used for extracting one parameter) will be divided amongst the group for parallel development while the group works together on data collection efforts. Iteration will occur during this phase until computer vision specifications are met.

During the machine learning training phase of the project, the group will largely be working together on the collective problem of optimizing the RPE prediction model, as well as the backend setup as a whole. Early in this phase, the minimum viable product will be completed, where the RPE predictor can take a video input and deliver an RPE prediction output. Model training will iterate using validation techniques to optimize the chosen model's parameters.

When overall backend development is nearing completion, the group will begin working on the front-end integration for the system. This phase can be broken into 3 areas of work which can be worked on concurrently: back-end finalization, front-end design, and integration. This final milestone encompasses the end of the project development cycle.

2.2 Design tools, hardware, instrumentation

Table 3 - Software Tools

Category	Tool	Purpose / Justification
Programming & ML Frameworks	Python 3.11, PyTorch, TensorFlow	Core development environment and machine-learning frameworks for training and inference.

Computer Vision Libraries	MediaPipe, MMPose, OpenFace	Feature extraction from video: bar tracking, joint angles, facial strain, and breathing patterns.
Data Science Libraries	NumPy, pandas, scikit-learn, Matplotlib	Data preprocessing, numerical computation, and visualization of results.
Development Environments	PyCharm, VS Code, Jupyter Notebook	IDEs for code implementation, testing, and interactive analysis.
Version Control & Collaboration	GitHub Private Repository	Code management, issue tracking, and version history.
Documentation & Communication	Microsoft Word, Microsoft Teams / Discord	Report writing, meeting coordination, and team communication.
Statistical Tools	MATLAB (Queen's Academic License)	Statistical validation and model-performance analysis.

Table 4 - Hardware Tools

Equipment	Specification / Use	Justification
iPhone Camera (Existing)	1080 p / 60 fps video capture for lift recordings	Provides high-resolution, stable video input for computer-vision models.
Tripod (Existing)	Adjustable mount for iPhone camera	Ensures consistent camera angles and minimizes motion blur.
Lighting Kit	LED lighting for uniform illumination	Enhances visibility and reduces lighting variance in recordings.
External SSD / Cloud Storage	1 TB capacity for data storage and model backups	Prevents data loss and supports large video datasets.
Centre for Advanced Computing (CAC)	GPU-enabled compute cluster for model training	Provides high-performance computing resources without cost.

Table 5 - Development and Collaboration Tools

Tool / Platform	Category	Purpose / Description	Justification
GitHub Repository	Version Control & Project Management	Repository for code storage, version history, issue tracking	Ensures modular code development and transparent collaboration across group members.
Microsoft Word	Documentation & Report Writing	Collaborative platform for editing technical reports, maintaining formatting, and tracking revisions.	Enables consistent formatting and synchronized editing of all written deliverables.
Microsoft Teams	Communication & Coordination	Weekly progress meetings, file sharing, and project discussions.	Facilitates structured, real-time communication with supervisor and team members.
Discord	Rapid Communication	Informal coordination and quick updates during development or testing sessions.	Provides continuous communication channel for debugging or urgent decisions.

Microsoft Excel	Task Tracking & Dataset Management	Tracks data labeling progress, experimental parameters, and team task assignments.	Simplifies data management and improves transparency in task progress.
Jupyter Notebook	Experimental Development	Testing and visualization environment for prototyping machine-learning pipelines.	Enables rapid iteration and debugging during model experimentation.

Table 6 - Data and Knowledge Resources

Resource Type	Example / Source	Purpose / Description	Access / Notes
Primary Dataset (Self-Recorded Videos)	Recorded lifts of group members (Squat, Bench Press, Deadlift)	Core dataset for training and validating models using real movement data.	Pending GREB approval for ethical clearance.
Public Datasets	YouTube Fitness Pose Dataset, Human3.6M, AIHub Pose Data, Kaggle	Supplementary training data to pre-train models or augment dataset if GREB approval is delayed.	Open-source and accessible online.
Framework Documentation	MediaPipe, MMPose, OpenFace,	Reference materials for model architecture, API usage, and parameter tuning.	Freely available via official developer websites.
LGBM Regressor	Microsoft LightGBM Library	Gradient boosting framework used for final RPE prediction by combining outputs from submodels (bar speed, pose, facial strain).	Open-source under MIT License, installed via lightgbm Python package.
Academic References	IEEE Xplore, Frontiers in Robotics and AI, PubMed	Scholarly resources supporting theoretical background and prior research.	Accessible through Queen's University library subscriptions.
Institutional Resources	Queen's Centre for Advanced Computing (CAC), Sci'65 Lab	Provides GPU computing resources and lab facilities for model training and testing.	Free institutional access through the ECE department.
Online Learning Resources	GitHub repositories, Stack Overflow	Practical implementations help for troubleshooting or optimizing frameworks.	Publicly accessible open-source learning materials.

2.3 Validation and Testing

Table 7: Required test criteria

Requirement / Spec	Test Method	Measurement / Metric	Target Value	Justification/Rationale	Responsible

					Member(s)
2.3.1 OpenFace Module					
Face Detection Reliability	Test A: Process 1080p/720p videos at direct & 45° angles.	Detection Rate: (Frames detected) / (Total frames)	≥95% detection	Ensures 95/100 frames are captured, providing sufficient data per rep.	Luke / Avedis
Landmark Stability (Jitter)	Test B: Record & analyze a still, neutral face.	Standard Deviation of (x, y) landmark coordinates.	≤2 pixels	A 2-pixel jitter is imperceptible on 1080p/720p video.	Luke / Avedis
Landmark Dynamic Motion	Test B: Record & analyze exaggerated facial expressions.	Distance between landmark pairs.	>50% change from rest	A large, non-trivial change (like 50%) confirms landmark motion is being captured correctly.	Luke / Avedis
AU Baseline (Noise Floor)	Test C: Process video of subject at rest between sets.	AU Intensity (e.g., AU04_r, AU07_r)	≤0.2	Defines a quantitative 'noise floor.' OpenFace AUs are 0-5; a 0.2 baseline ensures a neutral face is not misinterpreted.	Luke / Avedis
AU Strain Correlation	Test C: Compare AU intensities from low/high effort sets.	Proportionality & Visual Check	≥95% visual correspondence	Over 95/100 visible strain events are captured by the data. This confirms AUs are a reliable, high-fidelity feature for predicting RPE.	Luke / Avedis
2.3.2 MMDetection Module					
Object Detection Stability	Test A: Visual inspection & plot y-coord from 10-rep video.	Tracking Success Rate & Signal Quality	≥99% & Smooth Wave	Tracking must be continuous. 99% ensures the tracker doesn't lose the bar during a rep, which corrupts all timing/speed.	Rhys / Ajmal
Repetition Count Accuracy	Test B: Compare peak-finding algorithm to manual ground truth.		±1 rep per set	Accounts for non-exercise movements like re-racking or 'half-reps' at the end of a set. A ±1 tolerance is robust and realistic.	Rhys / Ajmal

Repetition Timing Accuracy	Test B: Compare algorithm's rep duration (sec) to ground truth.	Mean Duration Error	≤5%	A 5% error on a 3-second rep is only ±0.15s. High level of precision is required to detect subtle slowing.	Rhys / Ajma l
Bar Path Deviation	Test C: Compare std dev(x-coord) of Rep 1 vs. Final Rep.	x_dev_final vs. x_dev_rep1	Measurabl e Increase	A measurable increase validates the system's sensitivity to detect form breakdown.	Rhys / Ajma l
Bar Instability (Jitter)	Test C: Compare std dev (x-coord) of Rep 1 vs. Final Rep.	rmse_final vs. rmse_rep1	Measurabl e Increase	A measurable increase in jitter validates the system's sensitivity to detect instability.	Rhys / Ajma l
2.3.3 MMPose Module					
Joint Detection Coverage	Test A: Process videos for all 3 lifts.	Detection Rate: (Frames w/ all joints) / (Total)	≥95%	95% coverage ensures we have a complete dataset for analysis.	Luke / Aved is
Joint Detection Confidence	Test A: Average confidence scores from MMPose output.	Average Confidence Score	>0.8	A 0.8 threshold ensures we use only high-quality keypoints. Using low-confidence guesses would corrupt all angle calculations.	Luke / Aved is
Static Angle Accuracy	Test B: Compare calculated angle to manual protractor (all 3 lifts).	Angular Error	≤±4.5 degrees	A 5° tolerance is precise enough to meaningfully differentiate good form from poor form.	Luke / Aved is
Dynamic Form (ROM)	Test C: Compare key angles (Rep 1 vs. Final Rep)	Angle Change (e.g., min_knee_angle)	>3-degree difference	A 3° change is a small, subtle shift in ROM. Detecting it proves the system is precise enough to find fatigue-based form changes.	Luke / Aved is
2.3.4 LGBM Regressor					
Baseline Performanc e	Test A: Train/test DummyRegressor.	Baseline MAE	(Produces benchmar k)	Creates a quantitative baseline.	Rhys / Ajma l

RPE Prediction Accuracy	Test B: Evaluate final LGBM on 20% hold-out test set.	Mean Absolute Error (MAE)	≤0.5	0.5 is the smallest meaningful increment on the RPE scale, so meeting this target means the model predicts with human-level accuracy.	Rhys / Ajma 1
Model Robustness	Test C: Run 10-fold cross-validation.	Std. Dev. of MAE scores	≤0.15	A low standard deviation proves the model's performance is stable and not a 'lucky' result from one data split.	Rhys / Ajma 1
Feature Importance	Test D: Extract importances from trained LGBM.	Top 10 Features	Must include features from all 3 CV modules.	Critical sanity check that validates the project's core hypothesis that a combination of CV analysis is required to predict RPE.	Rhys / Ajma 1

2.3.1 OpenFace Validation for Facial Strain Analysis

Overall Objective: Verify OpenFace's ability to accurately process facial video data and extract key indicators of strain.

A. Input Data Quality Validation

Purpose: Identify how OpenFace performs with different video resolutions and camera angles. Confirm it can detect and track a face reliably under gym recording conditions.

Method: We will record short sample exercise videos under different recording setups:

1. **High-resolution (1080p)** camera facing the athlete directly.
2. **Medium-resolution (720p)** camera at a 45° angle.
3. **Low-resolution (480p)** camera positioned laterally.

We will process each video through OpenFace's face detection pipeline. Successful detection will be measured as the percentage of frames where a face bounding box is correctly identified and placed on the video.

Expected Result:

- A detection rate of ≥95% for high- and medium-resolution conditions indicate reliable handling of the video input data.
- A detection rate of ≤95% for the low quality (480p) shows which resolutions fall outside the system's operating limits.

B. Facial Landmark Tracking Validation

Purpose: Verify that OpenFace's landmark tracking accurately matches the athlete's real facial motion throughout each exercise repetition.

Method: We will record two sample videos of an athlete facing the camera for landmark tracking purposes.

1. **Stability Test:** The athlete will remain still with little to no facial motion and a neutral expression.
2. **Dynamic Test:** The athlete will perform slow, exaggerated facial actions like real physical exertion (furrowed brow, squeezed eyes, etc.), holding these expressions for at least 3 seconds.

These videos will be used to analyze the (x,y) coordinates of the key landmarks and their movements. [3]

Expected Results:

1. **Stability:** The standard deviation of the (x,y) coordinates for the neutral frozen expression must be ≤ 2 pixels considering some noise.
2. **Dynamic:** The distances between landmark pairs should change predictably, and the visual landmarks should follow a smooth and consistent path, reaching a max and then returning to neutral levels.

C. Facial Action Unit Validation

Purpose: Verify that OpenFace's Action Unit (AU) intensity estimates respond accurately and consistently to facial strain induced by physical exertion**.**

Method: Short video samples will be recorded where an athlete performs controlled exercises at low, medium, and high effort levels. OpenFace will extract AU intensity data for key facial actions [3].

We will evaluate the extracted AU intensities based on expected patterns:

1. **Resting position:** AU intensities should remain near zero.
2. **Exercise intensity:** AU intensities should increase proportionally with exertion and peak during max effort.
3. **Repetition consistency:** AU patterns should rise and fall consistently across repetitions.
4. **Visualization check:** Confirm that AU spikes correspond to the correct muscle movements (e.g., AU4 and eyebrows lower, AU6 and cheek raise) [3].

Expected Result:

1. Resting baseline AU intensities ≤ 0.2.
2. Peak AU intensities increase with effort, with consistent patterns across repetitions.
3. Visual correspondence $\geq 95\%$ between observed facial strain and AU spikes.

2.3.2 MMDetection Validation for Bar Speed Tracking

Overall Objective: Verify MMDetection's ability to consistently detect and track the bar during the three core exercises, accurately count repetitions and measure the duration, and highlight bar path instabilities.

A. Object Detection Stability Check

Purpose: Confirm that MMDetection can consistently detect and track the bar throughout an exercise video without losing the object or making unstable bounding boxes.

Method: A sample video will be processed through the MMDetection pipeline under consistent recording conditions. The test will assess two stability factors within the experiment:

1. **Detection Consistency:** Visually confirm (through MMDetection's visualization tool) [5] that the generated bounding box is tightly enclosing the bar in every frame.
2. **Motion Signal Validation:** We will extract vertical position of the bounding box's center will be extracted for every frame and plotted against time [5].

Expected Result:

• The bounding box correctly tracks the bar in ≥99% of frames.

• Vertical motion signal forms smooth, periodic waves matching the peaks and bottoms of repetitions.

B. Repetition Counting and Timing Validation

Purpose: Validate that the tracking system accurately processes bar motion to detect repetitions and measure their timing compared to the ground truth.

Method: Once stable bar tracking is confirmed, the vertical motion signal extracted from the bounding box coordinates will be analyzed to detect repetitions.

1. **Ground Truth Annotation:** The team will manually review the sample videos frame-by-frame. For each repetition performed, we will record the exact frame number for the start (peak height), bottom (trough), and end (peak height) of the movement. These logs will act as the ground truth.
2. **Automated Rep Detection:** Implement a peak-finding algorithm [7] and smoothing filters as necessary to automatically detect the peaks (tops of the lift). A single rep will be defined as the time (or number of frames) between two consecutive peaks.
3. **Comparison:** Will compare the algorithm's detected rep count and rep durations directly against the ground truth logs.

Expected Result:

• Automated repetition count matches ground truth within ±1 repetition per video.

• Average timing deviation ≤5% between detected and manual repetition durations.

C. Bar Path Deviation and Instability Validation

Purpose: Validate that the system can detect bar path deviations and instability caused by fatigue when performing an exercise.

Method: Record a set for each exercise of an athlete starting from the first rep and performing their maximum repetitions (until failure).

1. **Bar Path Deviation:** Calculate the standard deviation of the x-coord of the first and last reps. An increase in horizontal deviation signifies an inefficient looping bar path.
2. **Instability:** We will calculate the error rate between the raw y-coordinate data and its own smoothed path for the first and last rep. An increase in error from jittering or shakiness of the bar will signify fatigue.

Expected Result: A measurable increase in both the x-coordinate deviation and the y-coordinate error in the final rep compared to the firs rep.

2.3.3 MMPose Validation for Body Posture

Overall Objective: Validate MMPose's ability to detect all key body joints, measure stationary joint angles, and track changes in range of motion (ROM) as an indicator of form breakdown.

A. Joint Detection Stability

Purpose: Confirm that MMPose can consistently detect the full body and key skeletal joints of the user throughout the video sequence.

Method:

1. Process a sample video for each of the three key lifts.
2. **Coverage:** For each video, we will extract the output data and calculate the percentage of frames that successfully identified all 12 key joints (6 pairs) [4].

3. **Confidence:** Calculate the average confidence scores (provided by MMPose) [4] for all key points across all frames.
4. **Stability:** Plot the vertical coordinate of the primary moving joint over time (hip for the squat and deadlift, elbow for the bench press).

Expected Results:

1. **Coverage:** ≥95% of frames have all key joints detected.
2. **Confidence:** Average confidence score for all key points must be ≥ 0.8.
3. **Stability:** The joint's y-coordinate plot must be a smooth, periodic wave matching the exercise movement.

B. Joint Angle Accuracy

Purpose: Validate that the angles calculated by MMPose are accurate compared to real-world ground truth measurements.

Method: Process three sample videos through the MMPose pipeline, one for each exercise, using still frames for analysis. For each video, we will measure the ground truth angle manually using a digital protractor.

1. **Squat (Knee Angle):**
    - **Poses:** Record the athlete holding a "half squat" (120 deg) and a "full squat" (80 deg).
    - **Corresponding Key Points:** Hip, Knee, and Ankle [4].
2. **Bench Press (Elbow Angle):**
    - **Poses:** Record the athlete holding the “Bar at Top” (170 deg) and "Bar at Chest" (90 deg) positions.
    - **Corresponding Key Points:** Shoulder, Elbow, and Wrist [4].
3. **Deadlift (Hip Hinge Angle):**
    - **Poses:** Record the athlete holding the "Standing" (180 deg) and "Floor" (90 deg) positions.
    - **Corresponding Key Points:** Shoulder, Hip, and Knee [4].
4. Team will create an algorithm that calculates the joint angles between the corresponding key points for each exercise.

Expected Result:

For all three tests, the calculated joint angle must be within ± 4.5 deg of the manually measured ground truth.

C. Dynamic Range of Motion

Purpose: Validate that our system can detect form breakdown and changes in Range of Motion (ROM) by comparing a fresh rep to a fully fatigued rep.
Method: Record a set for each exercise of an athlete starting from their first rep and performing their maximum repetitions (failure). Extract and compare the ROM from the first fresh rep and the final fatigued rep. This involves extracting key static angles such as the knee angle for squat, elbow angle for bench press, and hip angle for deadlift of the first and final rep.
Expected Result: When comparing the first and final reps, it's expected to detect a significant change in the ROM angle (>3 deg), indicating fatigue in moving the weight for the exercise.

2.3.4 LGBM Regressor

Overall Objective: Validate the performance and robustness of the trained LightGBM (LGBM) regressor model in predicting the final 1-10 RPE value using the combined output taken from the previously tested Computer Vision modules.

A. Data Splitting and Baseline Model

Purpose: Generate an 80/20 train-test split for the dataset and create a simpler "baseline" model. This baseline model will serve as our benchmark to compare with the LGBM model to see if it provides significant prediction gains.
Method:

1. Combine and assemble a labeled dataset, with the rows containing the output features from the three CV models (OpenFace, MMDetection, MMPose) and one ground-truth label, the user-reported RPE value.
2. Split this dataset into a training (80%) and test set (20%).
3. Train a simple regression model on the training set (e.g.: DummyRegressor from scikit-learn) [7]. This model will predict the mean RPE of the training data.

4. Test this model using the test set and record its Mean Absolute Error (MAE).

Expected Result: This test produces our baseline error, which will be used to evaluate the effectiveness of the LGBM.

B. Final Model Accuracy Validation

Purpose: Validate the trained LGBM model's prediction accuracy against the unknown 20% test set. Ensure this meets our original performance specification.

Method:

1. Train the LGBM model on the 80% training set and tune key LGBM hyperparameters (learning rate, number of leaves, max depth, number of decision trees) [6].
2. Use the fully trained model on the 20% test set.
3. Compare the model's predictions to the user's subjective ground truth labels.
4. Calculate the Mean Absolute Error (MAE).

Expected Results: The MAE must be $\leq \pm 0.5$ RPE points.

C. Model Robustness Validation (Cross-validation)

Purpose: Make sure the model's performance is stable and not overfitting to a particular split or iteration.

Method:

1. Do a 10-fold cross-validation on the whole dataset [7]. This will split the data 10 times, training 10 different versions of the model on 90% of the data and then testing it on the remaining 10%.
2. Cross-validation will produce 10 different MAE scores. We will use the scores to calculate the mean and standard deviation of these 10 scores.

Expected Result:

1. The mean MAE should be low and close to the result found in our first test of the LGBM.
2. Standard deviation must be low.

D. Feature Importance Validation Sanity Check

Purpose: Validate that the model is using a combination of features that includes all CV models to predict the RPE value.

Method:

1. After training the LGBM model on the 80% training set, extract the feature importance scores [6].
2. Log and analyze the top 10 most decisive features in RPE prediction.

Expected Result:

1. The top 10 features must include at least one feature from each CV model (OpenFace, MMDetection, and MMPose).
2. Some features must logically have higher importance (repetition duration, angular velocity, etc.).

2.4 Potential Problems and Mitigation Strategies

Delay in GREB ethics approval for data collection due to technical issues with the GREB website. However, after reading the TCPS regulations, the project is classified as minimal risk and should not require any GREB approval that was initially proposed. As per TCPS article 2.5, the GREB involvement for this project falls most closely under Quality Assurance (QA) but is not required.

The GREB QA form for this project was submitted on October 27, 2025, as a due diligence measure, allowing GREB to provide any recommendations related to participant safety or ethical considerations. In the extreme case where GREB denies us from recording our own weightlifting videos for the project, the use of public/open-source lifting datasets and videos can be used for the MLM.

Currently, the team is unsure how large the data set size needs to be to successfully meet the project specifications. This may create a barrier in the future if the size of the data is too small, preventing the model from successfully training. To mitigate this issue, we plan on using data augmentation. Yet, if the dataset is still too small, the recovery plan is to use cross-validation to validate the MLM.

We can augment the data by taking each recording and cropping out select reps in a weightlifting set. This will allow us to record most videos on high RPE sets and remove the last couple of reps, effectively creating a video of a lower RPE set and saving time in the data collection phase. Additionally, the method of cross validation can be used, as it does not require the dataset to allocate select videos for a validation set.

A foreseeable problem that this project can face is limited GPU/computing resources for model training. Limited GPU/computing resources will cost more project time to test and train the computer vision model. The team currently have access to their personal GPUs for this project. However, in the case of training and testing taking too long on these GPUs, there are two more options available.

To mitigate this issue, the team can train and test the MLM on the computers available on Queen's University campus. These computers, such as the ones in the Bain Lab or the Queen's Compute Clusters, contain powerful GPUs. If these GPUs are still too weak, the recovery plan would be to use premium cloud GPU services such as Google Colab Pro, AWS, or Azure. These cloud GPU services provide access to the top industry GPUs for MLM testing. The team has allocated more than enough budget to purchase these services for the project if needed.

Another issue down the road is miscalculating the time required for setting up the data preprocessing pipeline, such as facial strain analysis, pose estimation, and bar speed change. If this step takes more time than what the team allocates in the project timeline, it will cost project time in the training and testing phase.

To mitigate this, the team has already begun implementing and testing different reputable open-source computer vision tools such as MMPose, MMDetection, and OpenFace. The implementation of these reputable tools saves time, as the team does not need to create their own machine learning tools. If the current open-source tools do not provide sufficient outputs, the recovery plan is to try different open-source tools or modify current open-source tools to work for the project instead of building one from the ground up.

The final foreseeable issue is low model accuracy, which will cost project time to address. To help mitigate low model accuracy, the team will conduct research on multiple model architectures. This way the MLM is not limited to one architecture. Training the dataset on different architectures can help optimize model accuracy and prevent the MLM from being restricted to a suboptimal algorithm.

In the event where the model accuracy is still low even after trying architecture changes, the recovery is to spend more time tuning hyperparameters, simplify features, or rescope the MLM's output. Training on different hyperparameters can further optimize the model's learning process and performance. Simplifying features can help remove noise and improve model generalization. Rescoping the MLM's output is a last resort and can involve narrowing the prediction target. Estimating a categorical RPE range (low, medium, or high) instead of an exact value can help artificially increase model accuracy.

Table 8: Risks and mitigation strategies

Problem	Mitigation	Recovery Strategy
Delay in GREB ethics approval	GREB QA form	Use of public/open-source lifting datasets and videos
Data set size	Data augmentation	Cross validation
limited GPU/ computing resources	Bain Lab GPUs or Queen's Compute Clusters	Cloud GPU services
Preprocessing pipeline set up time	Implementing and testing different reputable open-source computer vision tools	Try different open-source tools or modify a current open-source tools
Low model accuracy	Research and test multiple model architectures	Simplify features, tune hyperparameters, or rescope the MLM output

3. Milestones/Division of Labor

Table 9: Milestones

No.	**Milestone**	**Due date**	**Responsible member(s)**

1	Begin data collection	Week 8	Rhys/Ajmal
2	Begin computer vision setup	Week 8	Luke/Avedis
3	**Initial computer vision completed**	Week 10	Luke/Avedis
4	**Data collection complete**	Week 11	Rhys/Ajmal
5	Begin machine learning model training	Week 11	Ajmal/Rhys
6	Minimum viable product complete	Week 12	Ajmal/Rhys
7	Begin front-end app development	Week 13	Avedis/Luke
8	**Finalize back-end development**	Week 14	Rhys/Ajmal
9	**Finalize front-end development**	Week 15	Avedis/Luke

In Table 9, major milestones are bolded while intermediate milestones are not. Major milestones mark the end of a phase of development.
Our first major milestone is the completion of initial computer vision setup such that video can be fed as input and we can receive metrics on bar speed, facial action, and joint angles as output. Shortly after, we will be finishing data collection, where we will have a complete dataset of weightlifting videos for each of the three lifts we are using, each labeled with their RPE.

The milestone for completion of back-end development encompasses the completion of training for the RPE predictor model, as well as its integration with previously completed computer vision processing for inputs. Once the back-end is complete, front-end development will be finalised in preparation for project showcase.

Labour is divided throughout our project amongst multiple co-executed tasks. For each milestone, a primary and secondary responsible team member has been designated, with the primary responsible for spearheading that work phase with assistance from the secondary member and the rest of the group (when applicable). Working on different problems concurrently allows the group to maximize our productive output while reducing situations where a single group is capable of delaying the critical path of development.

4. Budget

Table 10: Hardware Bill of Materials

Part Name	Description	Qty	Est. Unit Cost (CAD)	Est. Total (CAD)	Source / Supplier
iPhone Camera	1080p / 60 fps mobile camera used for video recording	1	$0	$0	Existing personal device
Tripod	Adjustable mount for iPhone camera	1	$0	$0	Already owned
Lighting Kit	LED ring light or portable soft box for consistent recording conditions indoors	1	$50	$50	Amazon / Best Buy
External SSD or Cloud Storage	1 TB SSD or cloud backup for training data and model storage	1	$100	$100	Amazon / Best Buy
USB-C / Lightning Adapter & Mount	Adapter or stand for connecting iPhone to tripod and data transfer	1	$30	$30	Apple Store / Amazon
Misc. Accessories	Lighting extension cables, power bars, backup storage devices	–	$30	$30	Queen's Campus Store / Amazon
Subtotal (Hardware)				**$210 CAD**	

Table 11: Software Budget

Software / Service	License Type	Cost (CAD)	Notes
PyTorch / TensorFlow / MediaPipe / MMPose / OpenFace	Open Source	$0	All under MIT or Apache 2.0 licenses
MATLAB	Queen's Academic License	$0	Covered by institution license
PyCharm / VS Code / Jupyter Notebook	Community / Free	$0	Free for students
GitHub Private Repository	Education License	$0	Free with GitHub Education Pack
Overleaf (LaTeX)	Academic Plan	$0	Free for Queen's students
Subtotal (Software)			**$0 CAD**

Table 12: Infrastructure and Cloud Resources

Resource	**Purpose**	**Cost Estimate (CAD)**	**Notes**
Centre for Advanced Computing (CAC) Access	GPU training and model optimization	$0	Covered by Queen's research infrastructure
Cloud GPU Credits (AWS / Colab Pro / Azure)	Backup training resource if local compute is limited	$150	Pay-as-you-go GPU time for short training sessions
Google Drive / Dropbox Storage	Backup for raw and processed videos	$20	Optional low-tier subscription if needed
Subtotal (Infrastructure)			**$170 CAD**

Table 13: Other

Resource	**Purpose**	**Cost Estimate (CAD)**	**Notes**
Personal Trainer for Data Labeling	To accurately label dataset	$80-100	Cost varies based on personal trainers hourly rate.
Subtotal (Infrastructure)			**$100 CAD**

Table 14: Total Estimated Budget

Category	**Estimated Total (CAD)**
Hardware	$210
Software	$0
Infrastructure / Cloud	$170
Other	$100
Grand Total (Estimated)	**≈ $480 CAD**

This project has been developed in a cost-efficient and accessible way. By utilizing an existing iPhone camera and tripod, our team eliminates substantial hardware expenditures while still being able to produce high definition 1080p video appropriate for computer vision processing. Most hardware expenses are merely a lighting kit and external storage, which ensure consistent data collection conditions and stable management of the dataset. All software packages, such as PyTorch, TensorFlow, and MMPose, are either open-source or under Queen's academic licenses and therefore have no cost. Model training will be

primarily conducted through Centre for Advanced Computing (CAC) GPU cluster by Queen's University or through online cloud GPU's. A contingency fund is also budgeted for cloud GPU credits in case additional computation resources are required. As a whole, the project will remain well below the $600 budgetary line, for an estimated total of $380 CAD, with some allowance provided for any technical or resource-based future expenditures.

5. Risk Mitigation, Environmental Impact, and Legal Considerations

Our main safety consideration for this project is the risk presented by our data collection process. Since we are generating our own dataset, we will be taking videos of ourselves lifting weights for use in the training of the machine learning model. Lifting weights is a potentially hazardous activity with opportunity for common sports injuries. Using the ARECCI screen tool recommended by the GREB, this data collection phase is considered minimal risk and does not require GREB approval. To err on the side of caution regarding liability for the safety risks of this project, we are still completing a quality assurance application with the GREB. Input from the GREB will help us minimize health and safety risks.

The environmental impact of our project primarily presents itself in the use of cloud-computing GPU resources. These processors take a significant amount of energy and water resources. Our project has modest requirements for these resources, not requiring much processing time to complete the training phase of our project. While the overall environmental impact of the project is low due to its small scale, it is still important to be efficient in our use of computing resources.

6. Progress to Date

Initial Infrastructure Setup:

- **Repository Initialization:** The project repository was initialized on GitHub and structured in preparation for full AI-tool implementation and version control. This

included setting up the necessary directories for external tools, data collection, and source code.

- **OpenFace import:** Integrated OpenFace as a submodule in the project repository and configured the pipeline to utilize the package's functionality effectively.
- **Software Documentation:** The project's documentation and development environment was initialized and is in early development stages, including core library requirements in requirements.txt and a README.md.

Core Component Construction and Simulation:

With the infrastructure complete, the facial strain analysis module using OpenFace is progressing through the early stages of development.

- **OpenFace submodule construction:** The primary script, extract_pipeline.py, performs some video preprocessing and executes the OpenFace binary to generate AU data. Some experimenting with an interpolation function to ensure the AU data frames align with the video frame count, addressing data synchronization issues.
- **Simulation and Testing:** A visualization tool was made to simulate the model's output in real-time. This simulation uses OpenCV to display the video and dynamically overlays the facial tracking and AU values frame-by-frame.

7. Conclusion

RPEasy is an image-based system that estimates the Rate of Perceived Exertion (RPE) of an athlete from video recordings of core lifts using bar tracking, pose estimation, and facial strain analysis. These are combined with a LightGBM regressor to provide an objective, data-driven RPE score. Clear performance goals have been set, including ±0.5 accuracy for RPE, ≥95% feature detection reliability, and ≤0.5 s interface lag, to achieve both speed and accuracy.

Much progress has been made in the direction of a functional prototype. The GitHub repository and documentation are completely established, the OpenFace module and visualization pipeline are operational, and a general framework for testing OpenFace, MMPose, MMDetection, and the LightGBM regressor has been implemented. The project

timeline follows an iterative development cycle with weekly sprints and review sessions based on milestones, so the team is able to tighten up each module while continuously making progress towards unification.

The project is proceeding and well short of budget (≈ $380 CAD), leveraging existing recording equipment, open-source software, and Queen's Centre for Advanced Computing (CAC) GPU resources. Anticipated risk, e.g., lack of compute horsepower, dataset size, or timing out on ethics approval, has been identified, with mitigation in place.

Overall, the team is well-prepared to achieve its objectives. In the short term, the team will focus on completing data gathering, integrating pose and bar-tracking, and training the model to achieve the target accuracy. With the right technical support, ethical practices, and smart use of resources, RPEasy is set to provide a strong and creative solution that connects AI with strength training for valuable athletic insights.

References

[1] S. C. Glass and D. R. Stanton, “Self-Selected Resistance Training Intensity in Novice Weightlifters,” *The Journal of Strength and Conditioning Research*, vol. 18, no. 2, p. 324, 2004. Self-selected resistance training intensity in novice weightlifters – PubMed

[2] L. Bell, A. Ruddock, T. Maden-Wilkinson, and David. Rogerson, “Overreaching and overtraining in strength sports and resistance training: A scoping review,” Journal of Sports Sciences, vol. 38, no. 16, pp. 1897–1912, Jun. 2020. Overreaching and overtraining in strength sports and resistance training: A scoping review – PubMed

[3] T. Baltrusaitis, P. Robinson, and L. P. Morency, "OpenFace 2.0: Facial behavior analysis toolkit," 2018 IEEE 13th International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018.

[4] OpenMMLab, "MMPose: An open-source toolbox for 2D/3D human pose estimation," 2020. [Online]. Available: https://mmpose.readthedocs.io/en/latest/

[5] OpenMMLab, "MMDetection: An open-source object detection toolbox," 2019. [Online]. Available: https://mmdetection.readthedocs.io/en/latest/

[6] M. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," 31st Conference on Neural Information Processing Systems (NIPS), 2017.

[7] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.