Project Synopsis

Anomaly Detection in Exam Hall

- **Members:**
    - **Naveen — 3RB23CS407 — Team Lead**
    - **Bhagyesh — 3RB23CS403**
    - **Vivekanand — 3RB23CS406**
    - **Abhinav — 3RB23CS400**
- **Project Title:** *Anomaly Detection in Exam Hall*

---

2. Problem Statement

Examinations demand high academic integrity, but manual invigilation suffers from limitations such as fatigue, limited visibility, bias, and the inability to accurately monitor multiple students simultaneously. As class sizes increase, human invigilators often miss suspicious activities in real time, and reviewing hours of CCTV footage becomes tedious and inconsistent. Institutions require an automated, objective, and scalable solution that can process exam hall video feeds, detect anomalous or suspicious behaviour (e.g., copying, signalling, unauthorized device usage), and generate verifiable evidence such as snapshots and activity logs. Such a system enhances fairness, reduces workload, and improves post-exam auditing while respecting privacy and institutional guidelines.

---

3. Objectives and Scope

Objectives

- Detect anomalous behaviours in exam hall videos using a CNN-based classification model.
- Generate timestamped snapshots and structured anomaly logs for auditing.
- Provide a review interface for browsing incidents, visualizing metrics, and exporting summaries.
- Support continuous model improvement through retraining with newly labelled datasets.

Scope

- Complete end-to-end pipeline: video ingestion, pre-processing, model inference, anomaly detection, snapshot generation, and structured reporting.
- Modular architecture, including:
    - `video_analyzer.py` — core analysis engine
    - `model_trainer.py` — training and evaluation pipeline
    - `streamlit_app.py` — optional reviewer UI

- Deliverables include:
    - `anomaly_report.json` (incident log)
    - `anomaly_snapshots/` (evidence images)
    - `classification_report.txt`
    - `confusion_matrix.png`

---

4. Methodology (Process Description)

Data Preparation

- Collect and label exam hall video footage distinguishing normal vs. suspicious activity.
- Define annotation guidelines to ensure consistent labelling and ethical handling of sensitive data.
- Split data into training, validation, and test sets to prevent data leakage.

Model Development

- Use a CNN-based classifier (TensorFlow/Keras).
- Apply transfer learning with architectures such as MobileNetV2 or ResNet50 for improved accuracy and faster convergence.
- Pre-process frames by resizing, normalization, and batching.

Pipeline Flow

1. Ingest video input
2. Extract frames/time-based segments
3. Pre-process inputs
4. Run inference
5. Apply temporal smoothing and confidence thresholds
6. Flag anomalies
7. Save snapshots and logs with timestamps and metadata

Evaluation

- Use metrics such as Precision, Recall, F1-score, and ROC-AUC.
- Analyse confusion matrix to identify common misclassifications.
- Perform ablation testing (lighting, occlusion, seating density).

Recommended Diagrams

- Data Flow Diagram
- UML Use Case Diagram
- Activity/Sequence Diagrams

---

5. Hardware & Software Used

Hardware

- Desktop/Laptop with minimum 8 GB RAM
- GPU recommended (e.g., NVIDIA GTX 1060 or higher)
- Adequate storage for video datasets

Software

- Python 3.8+
- TensorFlow/Keras
- OpenCV
- NumPy, Pandas, Matplotlib
- Streamlit (for review UI)
- Virtual environment: `classroom_monitor_env`

---

6. Application and Future Scope

Applications

- Post-exam auditing with evidence-backed incident summaries
- Institutional integrity monitoring
- Standardized documentation for examination committees
- Useful dataset and pipeline for computer vision research
- Training resource for proctoring staff

Future Scope

- Expanded behaviour categories (signalling, device usage, unauthorized materials)
- Use of sequence models (LSTMs/Transformers) for better temporal analysis
- Privacy features such as face blurring or data anonymization
- Dashboard for multiple exam sessions and analytics
- Cloud-based or containerized deployment for large-scale use

---

7. Project Timeline (Gantt Overview)

Week	Task
1–2	Requirement analysis, literature survey, risk assessment, dataset planning
3–4	Data collection & labelling, pre-processing pipeline setup
5–6	Model training (baseline + transfer learning), hyperparameter tuning
7–8	Validation, metrics evaluation, confusion matrix analysis

Week	Task
9	Video analyser integration, anomaly handling, reporting format
10	UI development (Streamlit), PDF/CSV export features
11	Final documentation, testing, packaging, optional containerization

---

8. References / Bibliography

1. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, MIT Press, 2016.
2. R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.
3. F. Chollet, *Deep Learning with Python*, Manning, 2018.
4. OpenCV Documentation
5. TensorFlow & Keras Documentation
6. Kingma & Ba, Adam Optimizer, ICLR 2015
7. K. He et al., ResNet, CVPR 2016

Additional Notes

Resources & Limitations

- **Resources:** labelled video data, GPUs/CPUs, storage, annotation tools.
- **Limitations:** possible false positives/negatives, model dependency on high-quality datasets, privacy considerations, domain shift across exam halls.

Conclusion

This project proposes a robust, scalable, and evidence-based anomaly detection system for exam halls using CNN-based deep learning. The solution enhances fairness in academic evaluations by detecting suspicious behaviour objectively and consistently. It reduces manual workload, standardizes the auditing process with clear logs and snapshots, and supports transparent decision-making. The modular design ensures long-term maintainability, supports continuous model improvements, and provides relevant insights useful for both educational institutions and the broader EdTech ecosystem.