**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное автономное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ _____ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ_____

КАФЕДРА _____СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ_____

# РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## *К КУРСОВОЙ РАБОТЕ*

## *НА ТЕМУ:*

***Аналитический обзор систем и методов применения искусственного интеллекта для определения усталости человека с использованием мобильных вычислительных устройств***

Студент ____ИУ5И-31М___ _____ **Хаммуд Хала**
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы _____ **Ю.Е. Гапанюк**
(Подпись, дата) (И.О.Фамилия)

*2025*

**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное автономное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ
Заведующий кафедрой ___ИУ5____
(Индекс)
_____ ___В.И. Терехов__
(И.О.Фамилия)
«_08_» ____сентября_____ 2025 г.

# З А Д А Н И Е
## на выполнение курсовой работы

по дисциплине ____ НИР по обработке и анализу данных_____
Студентка группы _ИУ5И-31М__

_____Хаммуд Хала_____
(Фамилия, имя, отчество)

Тема курсовой работы _Аналитический обзор систем и методов применения искусственного интеллекта для определения усталости человека с использованием мобильных вычислительных устройств _____

Направленность КР (учебная, исследовательская, практическая, производственная, др.)
_____УЧЕБНАЯ_____
___
Источник тематики (кафедра, предприятие, НИР) _КАФЕДРА_____

График выполнения работы: 25% к ___ нед., 50% к ___ нед., 75% к __ нед., 100% к ___ нед.

***Задание*** Провести обзор существующих методов искусственного интеллекта для оценки усталости человека.Анализировать мобильные вычислительные устройства, применяемые в данной сфере.Сравнить основные алгоритмы машинного обучения, используемые для этой задачи. Подготовить выводы и рекомендации по дальнейшему развитию технологии.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на 34 листах формата А4.

Дата выдачи задания «_24_» ____ декабря_____ 2025 г.

**Руководитель курсовой работы** _____ **Ю.Е. Гапанюк**
(Подпись, дата) (И.О.Фамилия)

**Студент** _____ **Хаммуд Хада**
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре

# СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

## 1.1 Актуальность проблемы

В современном мире проблема усталости человека приобретает всё большее значение, особенно в таких областях, как транспорт, медицина, промышленность и IT-сектор. Длительная работа за компьютером, вождение автомобиля или выполнение сложных профессиональных задач может приводить к умственной и физической усталости, что негативно сказывается на продуктивности, качестве принимаемых решений и безопасности.

Развитие искусственного интеллекта (ИИ) и методов машинного обучения открыло новые возможности для автоматического выявления признаков усталости по данным, получаемым от пользователя. Особый интерес представляют мобильные вычислительные устройства (смартфоны и планшеты), поскольку они доступны широкому кругу пользователей и позволяют реализовать мониторинг состояния в реальном времени за счёт встроенных камер и датчиков при минимальных требованиях к дополнительному оборудованию.

## 1.2 Цель и задачи исследования

Целью данной работы является аналитический обзор современных систем и методов применения искусственного интеллекта для определения усталости человека с использованием мобильных вычислительных устройств, а также разработка и описание прототипа решения для мобильной платформы.

Для достижения цели были поставлены следующие задачи:

1. изучить традиционные и современные подходы к оценке усталости человека;
2. проанализировать алгоритмы машинного обучения и нейросетевые модели, применяемые для выявления признаков усталости;
3. рассмотреть технические возможности мобильных устройств (камера, датчики, вычислительные ограничения) для реализации мониторинга;

4. провести сравнительный анализ подходов по критериям точности, вычислительной сложности, энергопотребления и применимости на мобильных устройствах;
5. разработать общую схему процесса анализа признаков усталости и определить направления улучшения (повышение точности, устойчивость к условиям освещения, обеспечение приватности);
6. реализовать прототип мобильного приложения и выполнить первичную оценку его работоспособности.

### 1.3 Структура работы

Работа включает введение, обзор методов определения усталости и источников данных, анализ возможностей мобильных устройств, описание проектирования и реализации прототипа, а также обсуждение результатов, ограничений выбранного подхода и направлений дальнейшего развития. В заключении сформулированы основные выводы по проделанной работе.

## 2. Методы и технологии оценки усталости человека

### 2.1 Традиционные методы оценки усталости

Исторически оценка усталости часто основывалась на субъективных методах: опросниках, самоотчётах, а также тестах на внимание и когнитивные функции. Данные подходы имеют ряд ограничений:

- результаты зависят от самочувствия, мотивации и честности испытуемого;
- проведение тестирования может занимать значительное время;
- чувствительность и воспроизводимость в реальных условиях (работа, вождение, длительная нагрузка) часто оказываются недостаточными.

Более объективный подход представлен медицинскими измерениями, например: анализ вариабельности сердечного ритма, уровня гормонов стресса (в т.ч. кортизола), реакции зрачка и др. Однако такие методы, как правило, требуют специализированного оборудования и не подходят для непрерывного мониторинга в повседневных условиях.

### 2.2 Применение искусственного интеллекта в данной области

Развитие методов искусственного интеллекта (ИИ) и машинного обучения позволило автоматизировать выявление признаков усталости по

наблюдаемым данным, снижая зависимость от субъективной оценки человека. Наиболее распространённые направления включают:

- **компьютерное зрение** — анализ мимики, частоты моргания, степени закрытия век, положения головы и других визуальных маркеров;
- **анализ речевых сигналов** — выявление изменений темпа речи, интонации и других характеристик;
- **биометрические сигналы** — анализ сердечного ритма, кожно-гальванической реакции и других физиологических параметров (при наличии датчиков/устройств);
- **поведенческие признаки** — изменение активности, рост числа ошибок, снижение скорости реакции при выполнении задач.

Следует учитывать, что качество работы ИИ-систем зависит от условий съёмки/измерения (освещённость, положение устройства, шум), а также от индивидуальных особенностей пользователя. Поэтому на практике могут потребоваться калибровка и адаптация алгоритмов под конкретные сценарии использования.

## 2.3 Основные подходы машинного обучения

Методы машинного обучения, применяемые для оценки усталости, условно можно разделить на следующие группы:

### 2.3.1 Классификационные модели

Используются для различения состояний, например «устал/не устал» или нескольких уровней усталости. На практике применяются SVM, Random Forest, Logistic Regression и др.

### 2.3.2 Нейросетевые модели

Подходят для анализа сложных пространственных и временных зависимостей:

- **CNN** — чаще применяются в задачах компьютерного зрения (лицо, глаза, поза);
- **RNN/LSTM** — используются для анализа временных рядов, речи и некоторых биометрических сигналов;
- **Transformer-модели** — перспективны для обработки последовательностей и мультимодальных данных.

#### 2.3.3 Гибридные и мультимодальные решения

Комбинируют несколько источников данных и/или моделей, например объединение визуальных признаков лица с физиологическими сигналами или поведенческими показателями. Такие решения могут повышать устойчивость и точность, однако часто требуют больших данных для обучения и оптимизации вычислений, что особенно важно при реализации на мобильных устройствах.

# 3. Мобильные вычислительные устройства для мониторинга усталости

## 3.1 Сенсоры и датчики

Современные мобильные устройства и носимая электроника (умные часы/браслеты) содержат набор датчиков, которые могут использоваться для оценки косвенных признаков усталости и сонливости. Наиболее распространённые источники данных:

- **камера (оптические данные)** — анализ лица, глаз, мимики и положения головы;
- **акселерометр и гироскоп** — оценка движений, позы, наклона головы, особенности моторики;
- **микрофон** — анализ речевых характеристик (при наличии сценария использования);
- **физиологические сигналы** (ЧСС/HRV, SpO2, EDA и др.) — чаще доступны через носимые устройства или специализированные датчики и могут дополнять анализ на смартфоне.

При практической реализации на мобильной платформе важно учитывать доступность датчиков на конкретных моделях устройств, качество сигнала, а также энергопотребление и требования к приватности.

## 3.2 Анализ изображений и видео с мобильных устройств

Одним из наиболее распространённых подходов является анализ видеопотока с фронтальной камеры. В качестве признаков могут использоваться:

- характеристики глаз (частота моргания, доля закрытия век, длительные закрытия),

- признаки зевоты и изменения мимики,
- оценка положения и наклона головы.

Для обработки могут применяться как классические методы компьютерного зрения, так и нейросетевые модели. На практике используются библиотеки и фреймворки (например, OpenCV, MediaPipe и др.), а также готовые мобильные решения для детекции лица и ключевых точек. Важными факторами являются устойчивость к освещению, положению камеры и частичным перекрытиям лица.

### 3.3 Программные решения (категории)

Существующие программные решения можно условно разделить на:

1. приложения, ориентированные на мониторинг сна и восстановления (по движениям, расписанию, физиологическим данным при наличии носимых устройств);
2. приложения и функции, направленные на регуляцию цифровых привычек (время использования, уведомления), которые косвенно могут снижать переутомление, но не измеряют усталость напрямую;
3. специализированные системы (в т.ч. для транспорта и безопасности), использующие компьютерное зрение и/или физиологические датчики для выявления сонливости и предупреждения пользователя.

## 4. Сравнительный анализ методов искусственного интеллекта

Классические алгоритмы (SVM, Random Forest, Logistic Regression) часто применяются при наличии заранее извлечённых признаков и ограниченных наборов данных. Нейросетевые модели (CNN, RNN/LSTM, Transformer) эффективны при работе с изображениями и последовательностями, однако требуют оптимизации для мобильных устройств (скорость, память, энергопотребление). Перспективным направлением являются мультимодальные и гибридные подходы, объединяющие данные камеры, сенсоров и (при наличии) физиологических сигналов.

В рамках прототипа используется локальная обработка на мобильном устройстве, поэтому особое внимание уделяется вычислительной эффективности.

## 5. Блок-схема процесса анализа признаков усталости

Процесс анализа может быть представлен следующими этапами:

1. **Сбор данных** (камера/сенсоры/микрофон при необходимости);
2. **Предобработка** (фильтрация, нормализация, стабилизация изображения/сигнала);
3. **Анализ ИИ** (ML/DL-модели или правила);
4. **Постобработка и вывод результата** (оценка уровня, уведомления, рекомендации).


Рисунок 1 — Блок-схема процесса анализа признаков усталости.

## 6. Практическая реализация (разработка прототипа Android-приложения)

### 6.1 Постановка задачи

Цель практической части работы — разработка прототипа Android-приложения, способного в реальном времени оценивать **признаки усталости/сонливости** пользователя на основе анализа изображения с фронтальной камеры. Подход является неинвазивным и может использоваться для непрерывного мониторинга без дополнительного оборудования.

В рамках реализации минимально жизнеспособного продукта (MVP) были поставлены следующие задачи:

- автоматический запуск фронтальной камеры при открытии приложения;
- обнаружение лица пользователя в кадре;
- извлечение и анализ мимических признаков (параметры глаз, выражение лица и др.), доступных через модуль детекции лица;
- отображение рассчитанного индикатора в пользовательском интерфейсе;
- обеспечение работы приложения на устройстве без подключения к интернету.

Особое внимание уделялось:

- стабильности работы видеопотока и интерфейса в реальном времени;
- совместимости с устройствами Android версии 10 и выше;
- эффективности обработки с точки зрения потребления ресурсов и времени отклика.

Выбор технологий был ориентирован на локальную обработку данных (on-device) без использования облачных вычислений. В дальнейшем планируется расширение функциональности за счёт подключения дополнительных источников данных (например, анализ голоса, данные датчиков движения), а также реализация уведомлений при достижении порогового уровн утомления.


Рисунок 2 — Блок-схема процесса оценки признаков усталости в приложении.

### 6.2 Инструменты и технологии

Для разработки прототипа использовались современные инструменты, обеспечивающие совместимость с актуальными версиями Android и возможность локального выполнения алгоритмов:

- **Платформа:** Android;
- **Язык программирования:** Kotlin;
- **UI:** Jetpack Compose;
- **Доступ к камере:** CameraX;
- **Анализ лица:** ML Kit Face Detection;
- **Сборка:** Gradle;
- **Контроль версий:** Git;
- **Тестирование:** Android Emulator и физическое устройство (Samsung Galaxy A32).

ML Kit Face Detection позволяет выполнять детекцию лица и получать набор ключевых характеристик (в т.ч. вероятность улыбки, открытость глаз, углы поворота головы) при локальной обработке без передачи изображений на сервер, что важно с точки зрения приватности.

| **Компонент** | **Инструмент** | **Назначение** |
|---|---|---|
| Платформа | Android | ОС для мобильного устройства |
| Язык программирования | Kotlin | Разработка логики и UI |
| UI-фреймворк | Jetpack Compose | Построение интерфейса |
| Камера | CameraX | Доступ к фронтальной камере |
| Анализ изображений | ML Kit (Face Detection) | Определение мимики и признаков усталости |
| Сборка | Gradle | Компиляция проекта |
| Контроль версий | Git | История изменений кода |
| Тестирование | Эмулятор + Galaxy A32 | Проверка работы приложения |

### 6.3 Метод расчёта индикатора усталости и обоснование выбора признаков

В рамках прототипа используется эвристический (базовый) индикатор, так как применяемый модуль ML Kit Face Detection предоставляет вероятностные оценки отдельных мимических признаков (например, степень открытости глаз и вероятность улыбки), но не выполняет медицинскую диагностику усталости. Поэтому в текущей версии приложения рассчитывается интегральный показатель, отражающий уровень выраженности визуальных маркеров, потенциально связанных с утомлением/сонливостью.

Для вычисления индикатора используются признаки, доступные напрямую из ML Kit:

- LeftEyeOpenProbability и RightEyeOpenProbability — оценка открытости глаз;
- SmilingProbability — оценка вероятности улыбки.

Сначала вычисляется средняя открытость глаз:

$$\mathbf{EyeOpen} = \frac{\mathbf{LeftEyeOpen} + \mathbf{RightEyeOpen}}{2}$$

Затем формируется итоговый индекс усталости (в диапазоне 0–100%) как взвешенная сумма двух компонент:

$$\mathbf{FatigueIndex} = (0.5 \cdot (1 - \mathbf{EyeOpen}) + 0.5 \cdot (1 - \mathbf{Smile})) \cdot 100$$

где Smile=SmilingProbability.

Таким образом, при более закрытых глазах и отсутствии улыбки значение индекса увеличивается, а при открытых глазах и выраженной улыбке — уменьшается.

Для интерпретации индекса в интерфейсе используются пороговые значения:

- FatigueIndex ≥ 70% — высокий уровень утомления (рекомендуется перерыв);
- 40% < FatigueIndex < 70% — средний уровень;

- FatigueIndex ≤ 40% — низкий уровень утомления.

Выбор указанных признаков обусловлен тем, что они:

1. доступны в режиме on-device без передачи изображения на сервер;
2. вычисляются в реальном времени и подходят для мобильного сценария;
3. позволяют построить минимально жизнеспособный прототип (MVP) и визуализировать результат пользователю.

Направления улучшения. Дальнейшее развитие метода предполагает повышение устойчивости и информативности оценки за счёт:

- перехода от мгновенной оценки к временным характеристикам: частота моргания и PERCLOS (доля времени с закрытыми веками);
- учёта положения головы (head pose) и длительных наклонов;
- сглаживания/фильтрации показателя по времени (устранение “скачков”);
- персонализации порогов и калибровки под пользователя;
- при необходимости — внедрения облегчённой модели (например, TFLite) для более точной оценки в сложных условиях (освещение, очки, частичные перекрытия лица).

## 6.4 Этапы реализации

Разработка прототипа мобильного приложения проходила поэтапно, с акцентом на последовательную реализацию функциональных блоков, их отладку и интеграцию. Каждый этап включал анализ технических требований, выбор оптимального решения, реализацию кода и тестирование на физическом устройстве. Ниже приведено подробное описание основных этапов.

- **Этап 1. Инициализация проекта и настройка среды**

На первом этапе в Android Studio был создан новый проект под названием **FatigueFaceAI2**, с шаблоном `Empty Compose Activity` и языком программирования Kotlin. Далее были выполнены следующие действия:

- Настройка `build.gradle.kts` для подключения необходимых зависимостей:
    - Jetpack Compose;
    - CameraX (core, lifecycle, view);
    - ML Kit: face-detection;

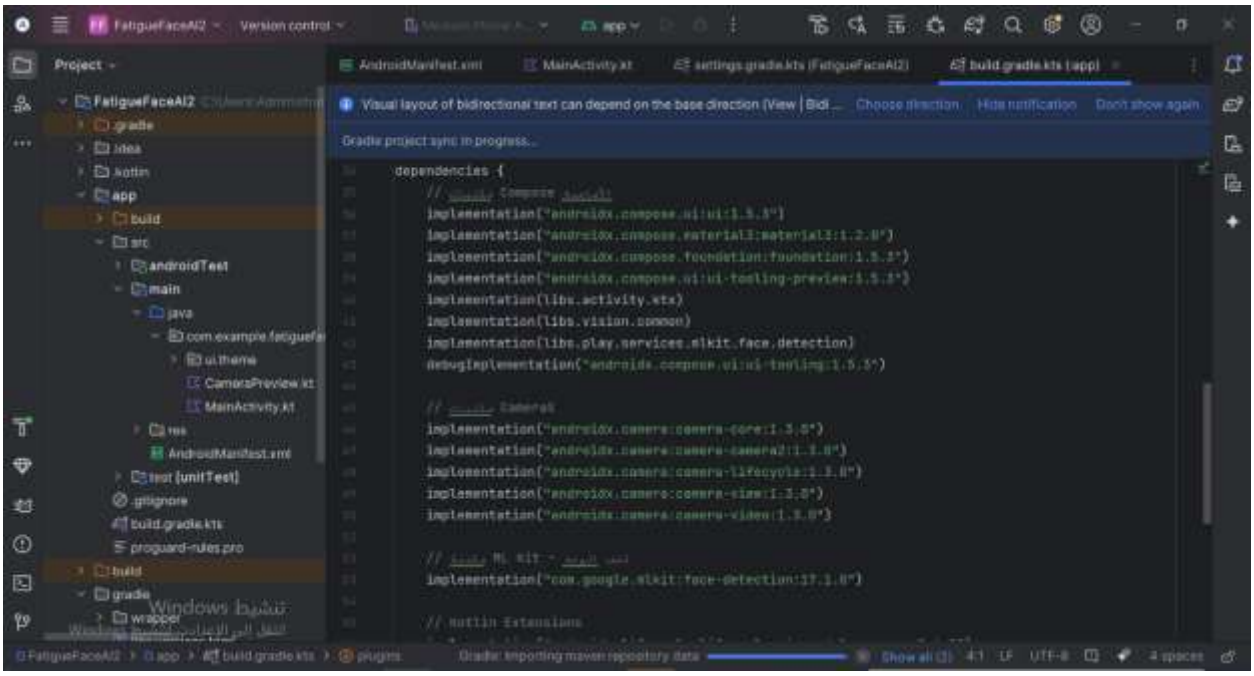



Рисунок 3 — Настройка зависимостей: Jetpack Compose и CameraX.

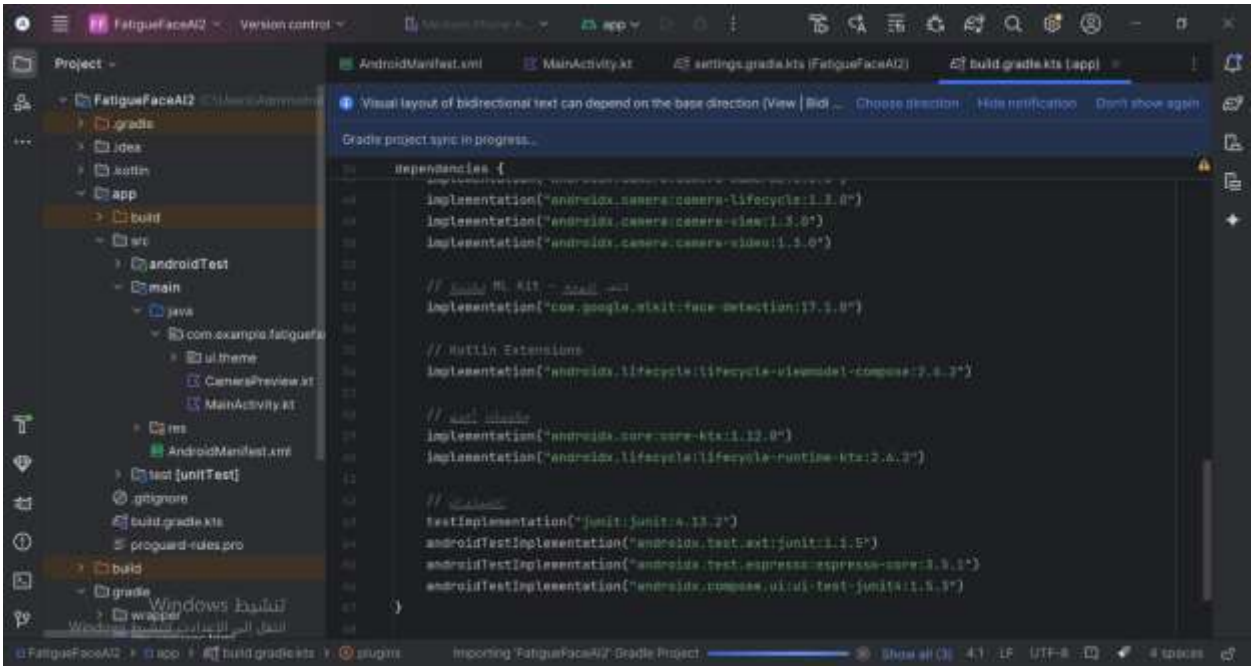


Рисунок 4 — Настройка зависимостей: ML Kit и другие библиотеки.

- Установка минимальной и целевой версии SDK;
- Создание структуры проекта, включая модули UI, логики и обработки изображений;
- Подключение темы оформления (Material 3) для унифицированного визуального стиля.

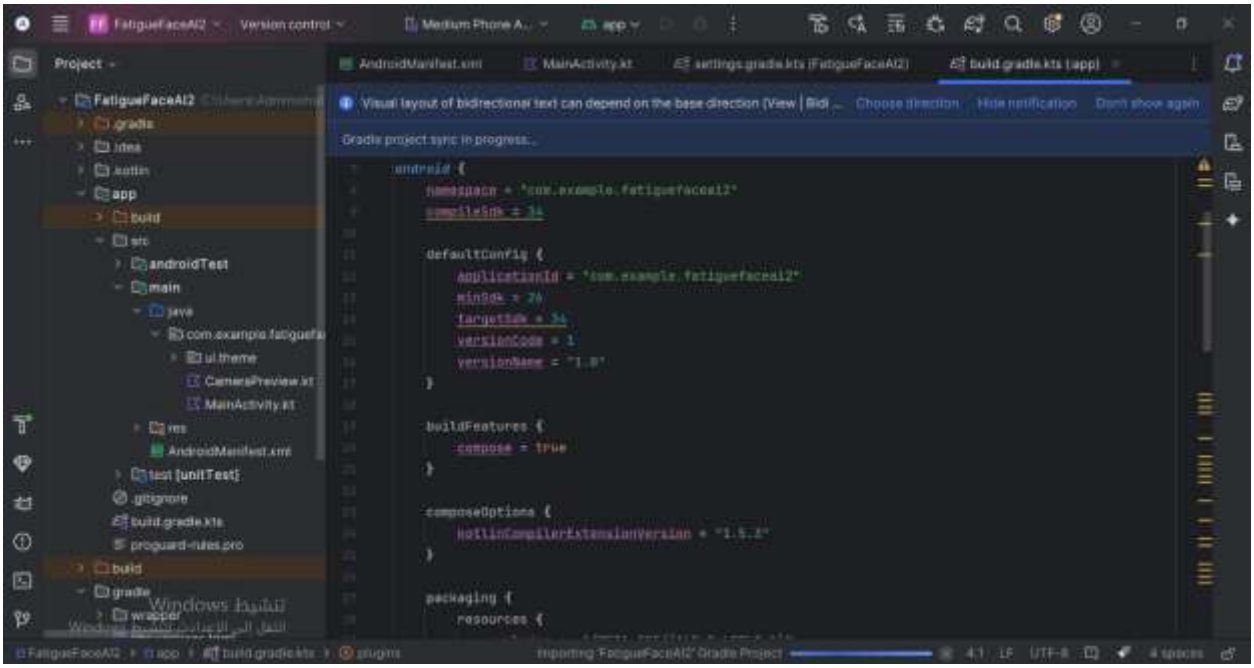



Рисунок 5 — Настройка compileSdk и minSdk в проекте FatigueFaceAI2.

Проект с самого начала разрабатывался с учётом требований к устройствам под управлением Android версии 10 и выше, с приоритетом на стабильную работу и локальное выполнение алгоритмов без необходимости в подключении к интернету.

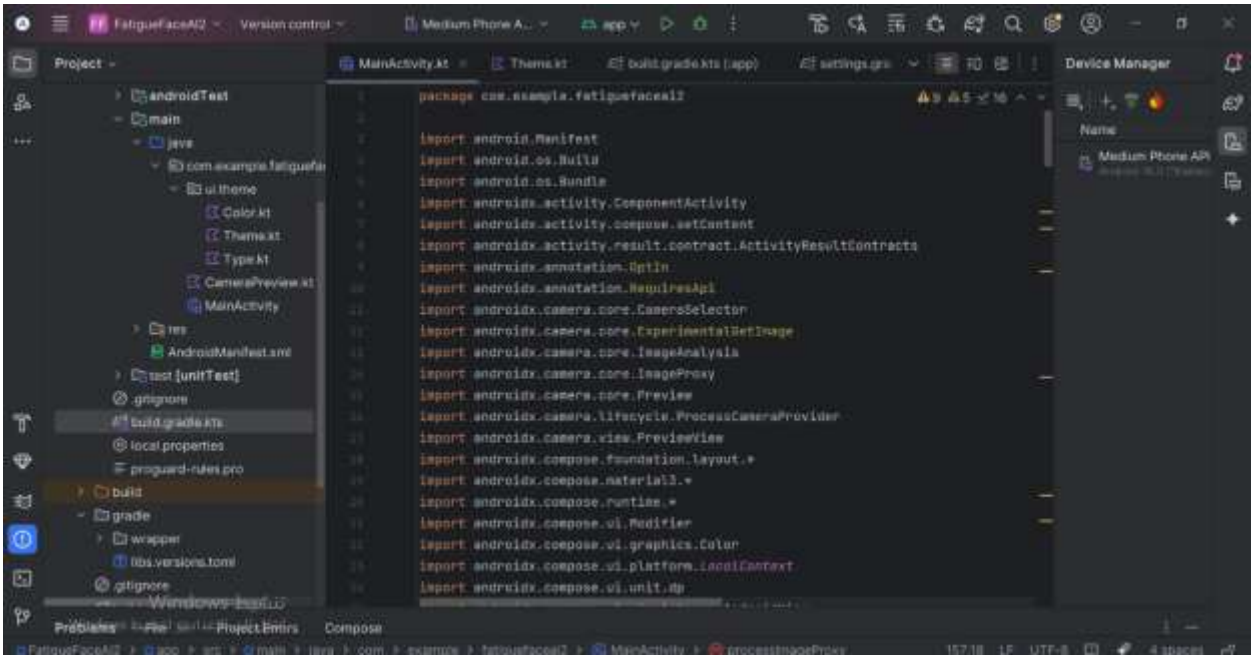
- **Этап 2. Работа с камерой — интеграция CameraX**

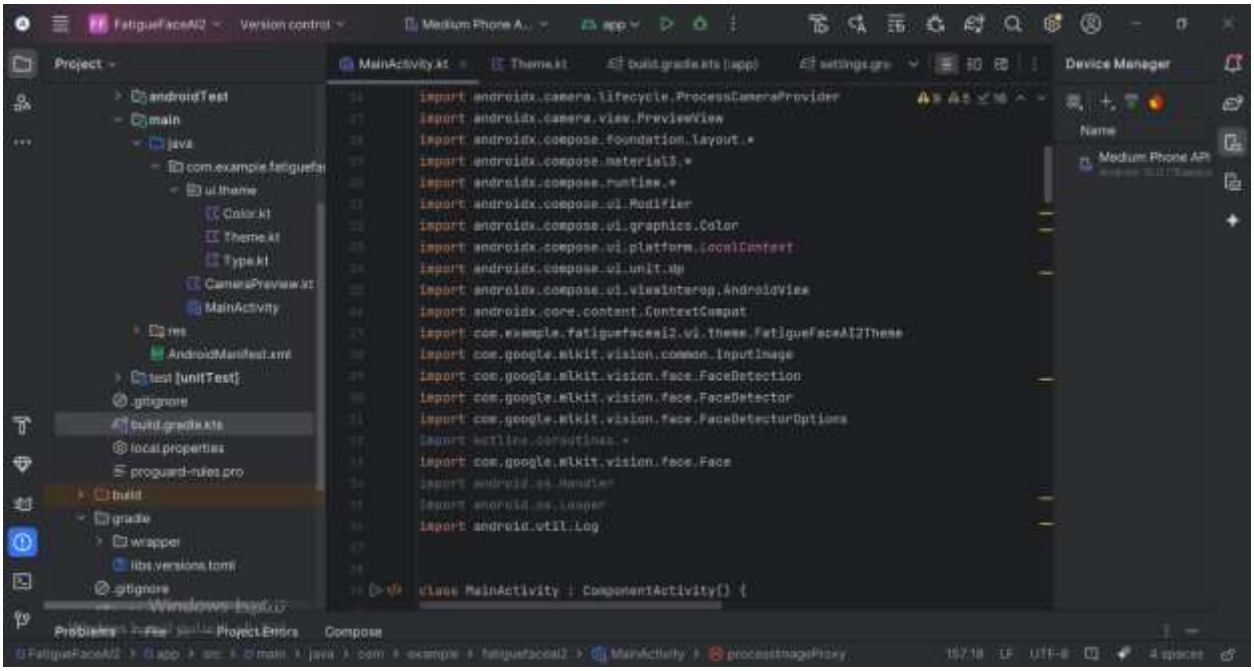
Следующим ключевым шагом стало подключение фронтальной камеры устройства. С этой целью был реализован компонент `CameraPreview`, использующий API CameraX:

- Реализован запрос разрешения на использование камеры через `ActivityCompat.requestPermissions`;
- В случае одобрения — запускается `PreviewView`, отображающий поток изображения ;

- Камера привязывается к жизненному циклу активити с помощью `bindToLifecycle`, что обеспечивает корректную работу при поворотах экрана и приостановке приложения;
- Предусмотрено автоматическое отключение камеры при закрытии приложения .

Пример кода:





Рисуноки 6,7 — Импорты CameraX в MainActivity.


Рисунок 8 — Запрос разрешения на использование камеры.

На рисунке 8 представлен скриншот с физического устройства, на котором приложение FatigueFaceAI запрашивает разрешение на использование фронтальной камеры.

Этот этап позволил добиться стабильного отображения видеопотока в режиме реального времени и стал основой для последующей обработки изображения.

- **Этап 3. Интеграция ML Kit Face Detection**

После успешной интеграции камеры началась реализация системы распознавания лица:

- Использован класс `InputImage` для преобразования кадра в подходящий формат;
- Настроен `FaceDetectorOptions`, включая режимы:
  - `PERFORMANCE_MODE_FAST` — для ускорения обработки;
  - `CLASSIFICATION_MODE_ALL` — для получения вероятностей улыбки и открытых глаз;
- Внедрён обработчик `ImageAnalyzer`, который анализирует каждый кадр и извлекает ключевые признаки (моргание, улыбка и т.п.);
- Добавлены лог-файлы и консольный вывод для отладки на ранних стадиях.

Пример вызова:

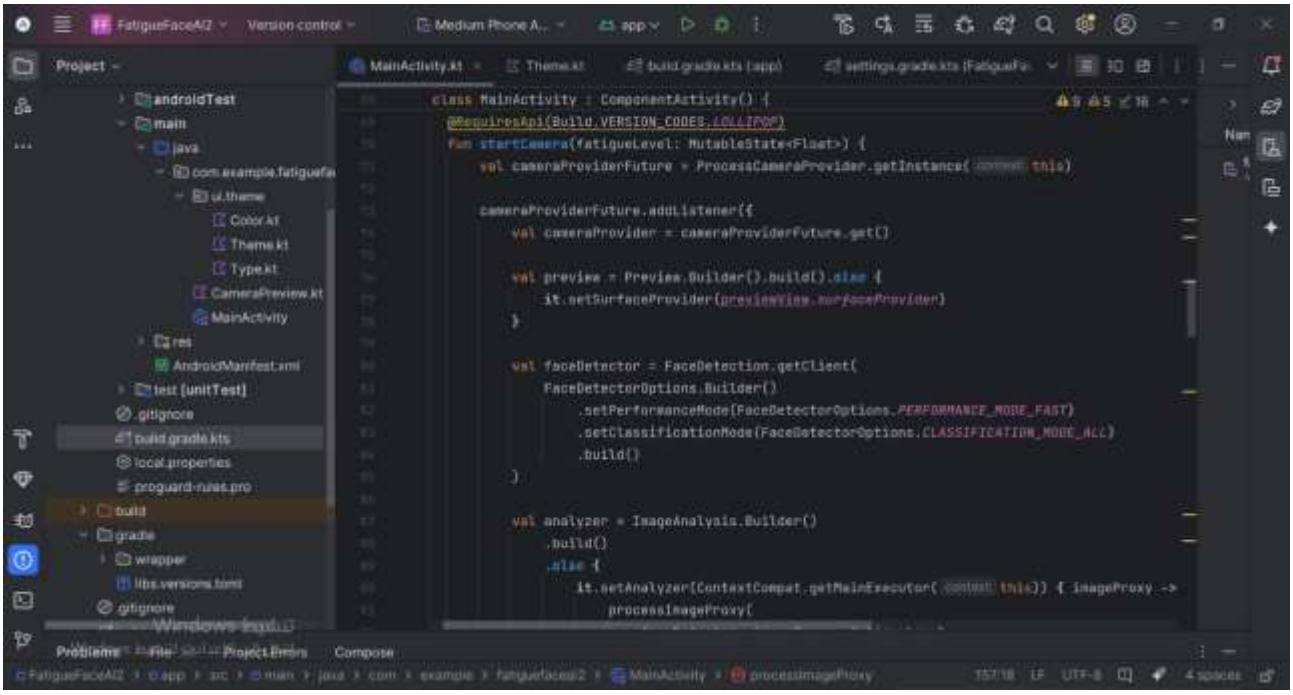
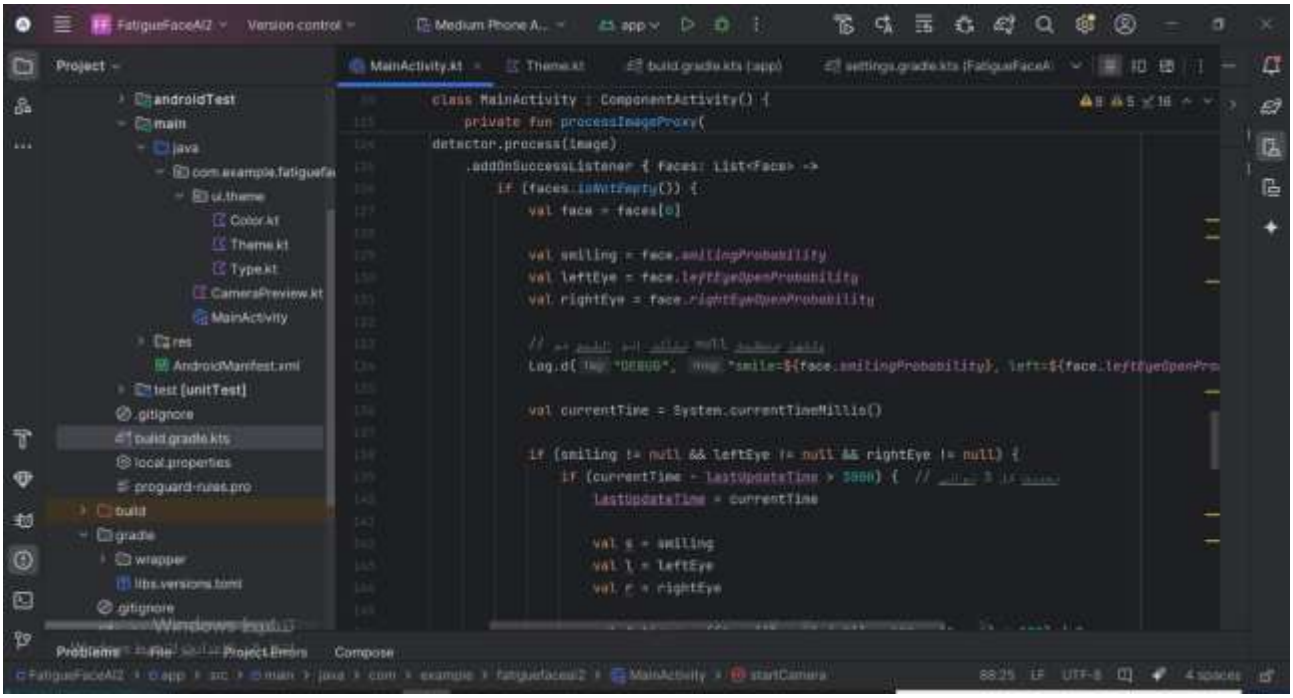



Рисунок 9 — Пример кода для анализа изображения с помощью ML Kit Face Detection.

## ○ Этап 4. Логика оценки усталости

На основе полученных признаков была реализована формула, позволяющая приблизительно оценить уровень усталости:



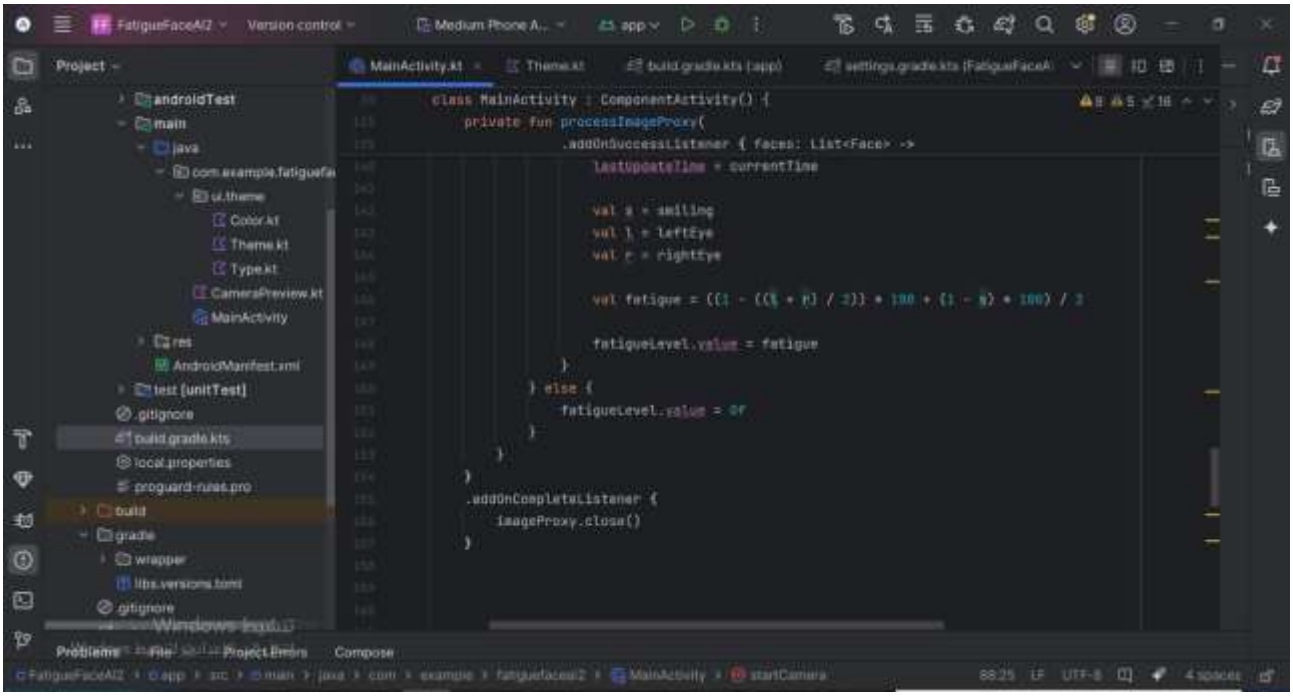


Рисунок 10,11 — Пример вычисления уровня усталости на основе мимических признаков (глаза и улыбка).

При этом:

- уровень усталости выше 70% помечался как высокий;
- ниже 40% — как допустимый.

Интерфейс динамически менялся: цвет текста и сообщение адаптировались под результат.


Рисунок 12. Фрагмент кода интерфейса: динамическое отображение уровня усталости пользователя с адаптацией цвета текста в зависимости от значения.

- **Этап 5. Интерфейс и обратная связь**

На этапе финальной интеграции реализован простой, но наглядный пользовательский интерфейс с использованием Jetpack Compose:

- Отображение видеопотока с камеры (`PreviewView`);
- Вывод текста с результатом анализа;
- Использование `Text`, `Column`, `Modifier` для построения структуры UI;
- Планировалось добавление индикатора (ProgressBar), но из-за ограничений времени не было реализовано в рамках текущего этапа.

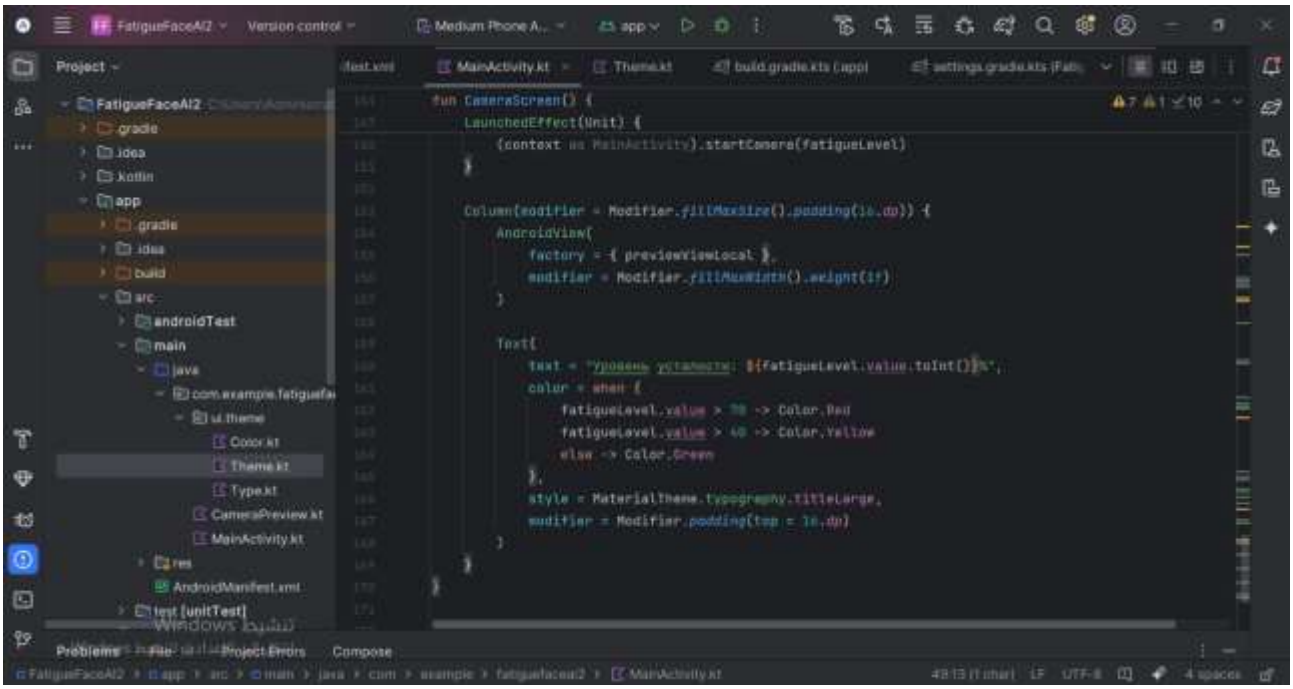



Рисунок 13. Реализация пользовательского интерфейса в Jetpack Compose: отображение видеопотока с камеры и уровня усталости.

## ◆ Итоги этапов

Все этапы разработки были направлены на реализацию базовой работоспособной версии системы. Основной функционал (камера + анализ лица + первичная логика оценки) был успешно протестирован на реальном устройстве. Интерфейс функционирует корректно, интерфейс отзывчивый, но дальнейшая доработка необходима для повышения точности и надёжности.

- **Пример кода**

В данном разделе представлены ключевые фрагменты кода, использованные в реализации прототипа. Каждый из них иллюстрирует важный этап — от анализа изображения до построения пользовательского интерфейса. Также даны пояснения к каждой строке для лучшего понимания логик.

- **Получение признаков усталости из изображения:**


Рисунок 13. Извлечение признаков усталости с использованием ML Kit: вероятности улыбки и открытых глаз.

**Пояснение:**

- ML Kit возвращает вероятности (от 0 до 1) для каждого обнаруженного признака.
- `smilingProbability` — вероятность того, что пользователь улыбается.
- `leftEyeOpenProbability, rightEyeOpenProbability` — вероятность того, что левый и правый глаз открыты.
- Используется оператор ?: 0.0f для защиты от null, если признак не обнаружен.

- **Расчёт уровня усталости:**

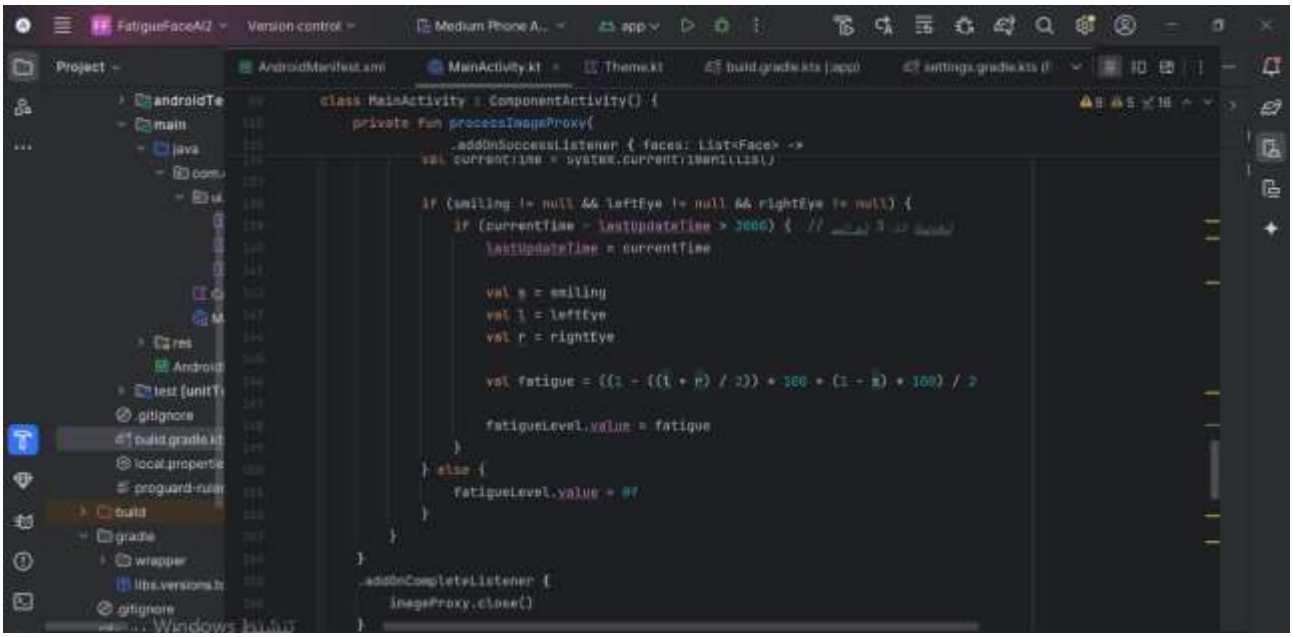



Рисунок 13. Расчёт уровня усталости на основе вероятностей улыбки и открытых глаз с использованием сбалансированной формулы.

**Пояснение:**

- Чем меньше вероятность открытых глаз, тем выше усталость.
- Чем меньше вероятность улыбки, тем выше вероятность утомления.
- Результат масштабируется до шкалы от 0 до 100.
- Формула сбалансирована по весу: 50% — глаза, 50% — улыбка.


Рисунок 14.

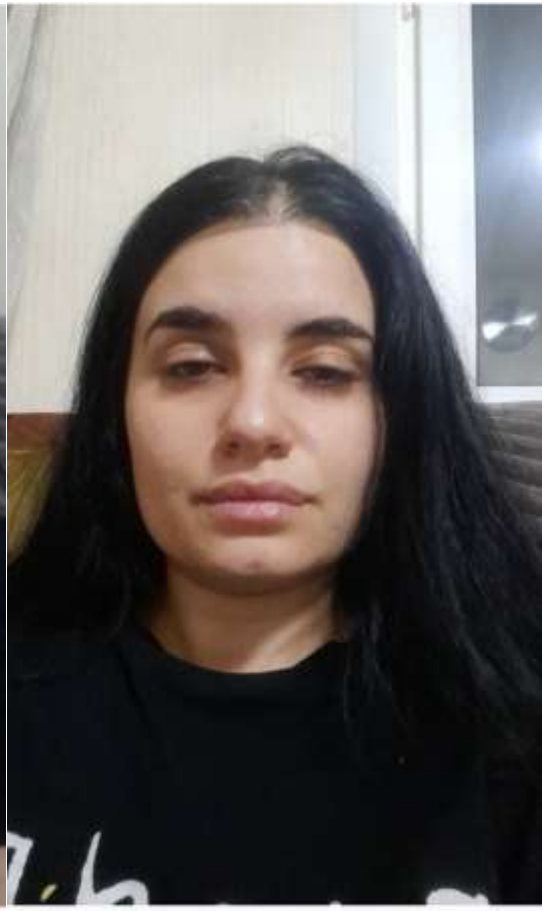



Рисунок 15.

На Рисунок 14.  видно бодрое лицо с открытыми глазами и лёгкой улыбкой. Уровень усталости пониженный — **0%**.

На Рисунок 15. показано уставшее лицо: глаза полуприкрыты, отсутствует улыбка. Система определила высокий уровень усталости — **97%**.

На Рисунок 16.  показано лицо пользователя в промежуточном состоянии усталости: глаза открыты, но отсутствует улыбка, в результате чего система

классифицировала уровень усталости как средний — значение отображено жёлтым цветом.

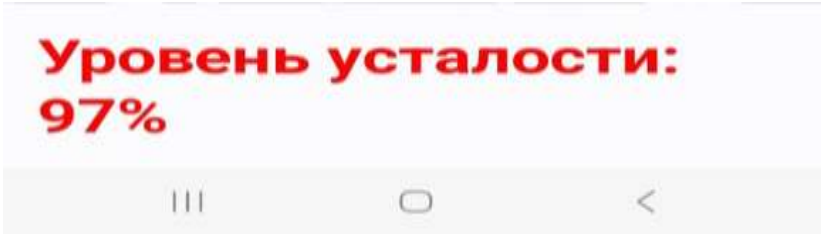

Рисунок 16.

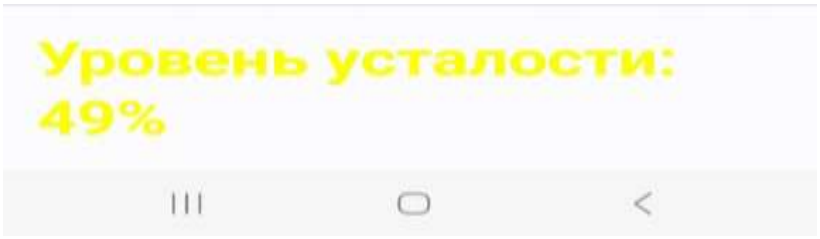
- **Отображение результата в интерфейсе:**

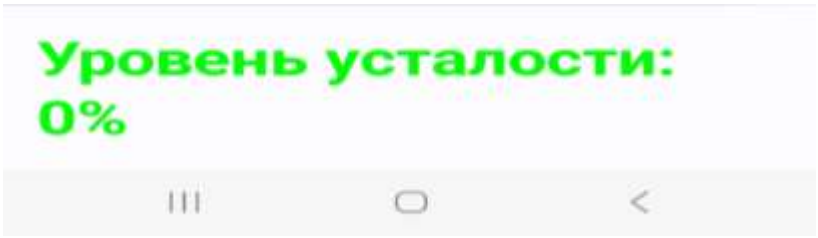

Рисунок 17. Динамическое отображение уровня усталости в интерфейсе с использованием компонента Text и цветовой индикации.

**Пояснение:**

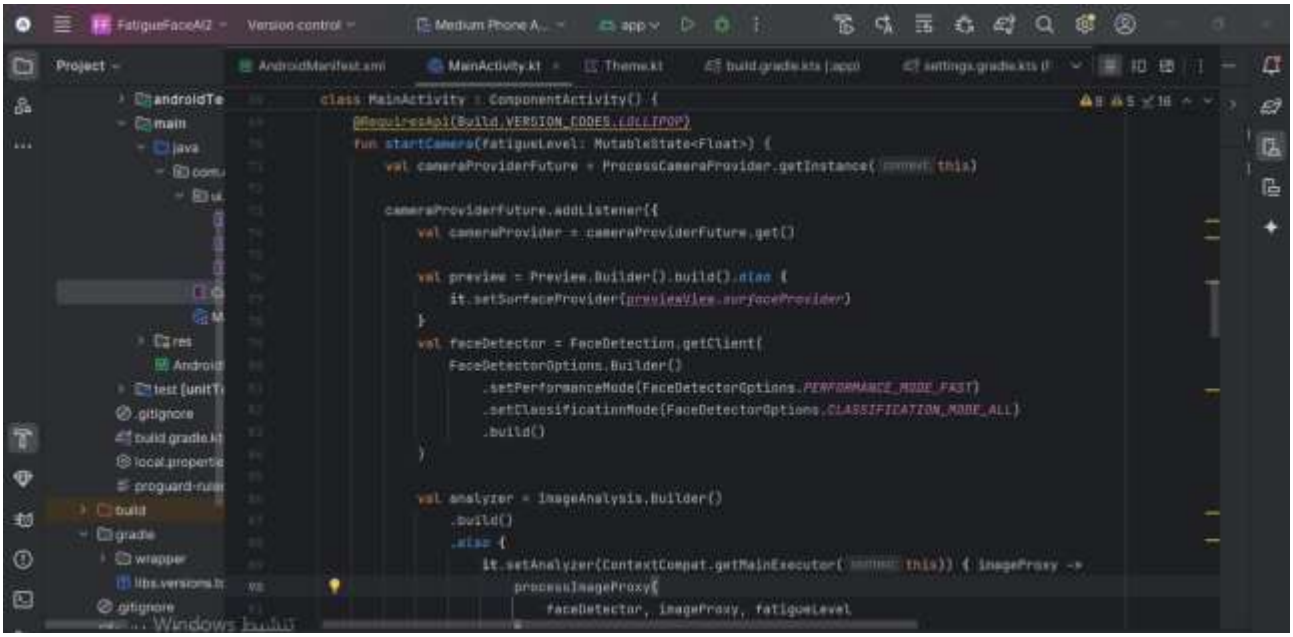
- Используется компонент `Text` из Jetpack Compose.
- Цвет текста меняется динамически в зависимости от уровня усталости:
  - Красный — критическая усталость.
  - Жёлтый — средний уровень усталости, частично выраженные признаки.
  - Зелёный — допустимое состояние.
- Результат округляется до целого числа.

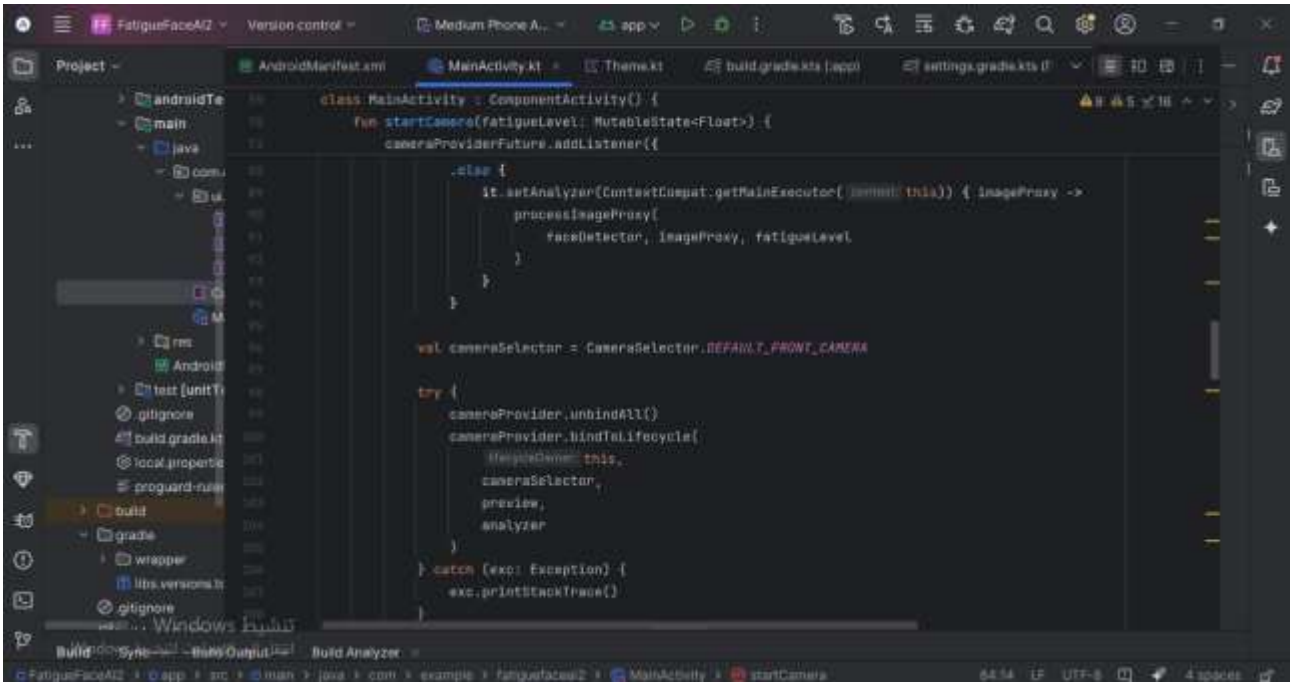





- **Встроенное окно камеры (PreviewView через AndroidView):**






Рисуноки 18,19. Инициализация камеры и интеграция компонента PreviewView через AndroidView для отображения видеопотока.

Изображение демонстрирует работу встроенного окна камеры в приложении: отображается видеопоток с фронтальной камеры в реальном времени.


Рисунок 20. Демонстрация работы встроенного окна камеры и отображения уровня усталости в пользовательском интерфейсе.

## 6.5 Экспериментальная оценка прототипа (результаты тестирования)

**Цель тестирования** — проверить работоспособность прототипа в реальном времени на мобильном устройстве, оценить устойчивость детекции лица и приблизительные показатели производительности (задержка обработки и частота кадров) в типичных условиях использования.

**Условия и оборудование.**
Тестирование проводилось на устройстве Samsung Galaxy A32 (Android 12), а также на Android Emulator (для первичной проверки). Приложение использует фронтальную камеру через CameraX и выполняет детекцию лица и извлечение признаков с помощью ML Kit Face Detection. Обработка выполняется локально (on-device), без передачи видеопотока на сервер.

**Сценарии тестирования.**
Проверка выполнялась в следующих режимах:

1. Хорошее освещение (дневной/яркий искусственный свет).
2. Слабое освещение (вечер/недостаточный свет).
3. Движение головы/повороты (умеренная динамика, имитация реального поведения пользователя).

**Оцениваемые метрики:**

- Устойчивость детекции лица — доля времени/кадров, когда лицо успешно обнаруживается (Face detection success rate).
- Производительность — ориентировочная частота кадров (FPS) и/или средняя задержка обработки одного кадра (мс/кадр).
- Стабильность работы — наличие зависаний, пропусков кадров, заметных задержек интерфейса.

**Примечание:** значения FPS/мс/кадр приводятся по результатам локального измерения в ходе тестирования прототипа. Если использована приблизительная оценка, это отмечается как «оценочно».

**Связь с демонстрационными примерами.**
Скриншоты (см. рисунки 14–16) демонстрируют работу алгоритма на различных входных условиях и соответствующее изменение вычисляемого индикатора усталости в UI.

### 6.6 Приватность и безопасность данных

Поскольку прототип использует фронтальную камеру, вопросам приватности и безопасности уделяется отдельное внимание. Обработка видеопотока выполняется локально на устройстве (on-device) с использованием CameraX и ML Kit Face Detection, без передачи изображений или видеоданных на внешние серверы и без использования облачных вычислений.

В рамках текущей реализации фото и видеозаписи не сохраняются: кадры используются только для анализа в оперативной памяти и отображения результата пользователю в интерфейсе. Доступ к камере осуществляется исключительно после предоставления пользователем соответствующего разрешения (Camera Permission) в соответствии с механизмами безопасности Android.

## Выводы

В результате проведённой практической работы была успешно реализована первая версия прототипа мобильного приложения, предназначенного для оценки уровня усталости пользователя на основе анализа изображения с фронтальной камеры. Несмотря на то, что полная логика анализа признаков усталости с применением нейросетевых моделей пока находится в стадии разработки, текущая реализация уже демонстрирует работоспособность ключевых компонентов системы.

Были достигнуты следующие важные результаты:

- реализовано стабильное подключение и отображение фронтальной камеры с помощью библиотеки CameraX;
- подготовлена архитектура для анализа изображения в реальном времени с использованием ML Kit;
- реализована базовая структура пользовательского интерфейса на Jetpack Compose, способная адаптироваться к изменениям данных;
- протестировано приложение на реальном устройстве Samsung — камера функционирует корректно, интерфейс работает плавно и стабильно;
- заложена основа для расчёта уровня усталости на основе вероятностей открытых глаз и улыбки, получаемых от Face Detection API.

Разработка данного прототипа показала, что современные мобильные устройства вполне способны выполнять задачи реального анализа выражения лица в автономном режиме, без подключения к облачным сервисам .Это делает возможным применение подобного подхода в таких сферах, как:

- контроль состояния водителей и операторов;
- интеллектуальные уведомления при переутомлении в офисной работе;
- образовательные приложения, отслеживающие внимание студентов;
- медицинские и реабилитационные платформы.

**Главный вывод** заключается в том, что интеграция искусственного интеллекта в мобильные системы мониторинга позволяет создавать персонализированные и чувствительные решения для оценки психофизиологического состояния пользователя.

Полученные результаты являются основой для дальнейшего расширения функционала приложения, включая:

- добавление аудиоанализа (оценка голоса и интонации);
- анализ движений головы и положения тела;
- обучение индивидуализированных моделей на основе накопленных данных пользователя.

Таким образом, проведённая работа подтверждает практическую применимость технологий компьютерного зрения и машинного обучения для создания интеллектуальных мобильных решений в области мониторинга усталости.

## Список использованных источников


1. Zhang Z., Zhang C.A Review of Fatigue Detection Based on Machine Learning Algorithms. – Journal of Healthcare Engineering, 2021.
2. Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press, 2016.
3. Hinton G., Salakhutdinov R. Reducing the Dimensionality of Data with Neural Networks. – Science, 2006.
4. Li G., Chung W. Smartwatch-Based Fatigue Detection System Using Machine Learning. – Sensors, 2022.
5. OpenCV Documentation. URL: https://docs.opencv.org
6. Android Developers – CameraX Guide. URL: https://developer.android.com/training/camerax
7. Android Developers – ML Kit Face Detection. URL: https://developers.google.com/ml-kit/vision/face-detection
8. Jetpack Compose Official Documentation. URL: https://developer.android.com/jetpack/compose
9. Kotlin Language Documentation. URL: https://kotlinlang.org/docs/home.html
10. Google ML Kit Overview. URL: https://developers.google.com/ml-kit
11. Gradle Build System Documentation. URL: https://docs.gradle.org
12. Lifecycle-aware Camera App with CameraX. URL: https://developer.android.com/topic/libraries/architecture/lifecycle
13. Compose + CameraX Integration Example. URL: https://github.com/android/camera-samples/tree/main/CameraXBasic
14. Google Face Detection API: Best Practices. URL: https://developers.google.com/ml-kit/vision/face-detection/android

15. Android Permission Handling. URL: https://developer.android.com/training/permissions/requesting