T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

LOCAL AUTHENTICATION WITH FACE DETECTION - IATTEND

UMUT AY ALPER

SUPERVISOR
PROF. ERKAN ZERGEROĞLU

GEBZE
2024

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

LOCAL AUTHENTICATION WITH FACE DETECTION - IATTEND

UMUT AY ALPER

SUPERVISOR
PROF. ERKAN ZERGEROĞLU

2024
GEBZE

GRADUATION PROJECT
JURY APPROVAL FORM

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 21/01/2024 by the following jury.

JURY

Member
(Supervisor) : Prof. Erkan ZERGEROĞLU

Member : Dr. Lecturer Member Alp Arslan BAYRAKÇİ

ABSTRACT

This graduation project proposes the implementation of an advanced mobile-based attendance tracking system using Flutter for university lecture halls and classrooms. The system integrates cutting-edge facial recognition technology and meticulous location verification processes to ensure precise attendance management.

In the proposed system, the instructor initiates the attendance process from their computer, prompting students to capture selfies using their mobile phones. A key distinguishing feature is the autonomous execution of the facial recognition process entirely within the mobile application, thereby minimizing any operational impact on the instructor's system.

The mobile application, developed exclusively with Flutter, exhibits a high degree of autonomy, handling intricate facial recognition tasks efficiently. Furthermore, the system employs rigorous location verification, cross-referencing the instructor's computer location as a benchmark to confirm the physical presence of students in the designated lecture hall or classroom.

A paramount aspect of the system involves secure user registration, requiring students to capture initial selfie photos during their inaugural use of the application. These photos serve as a foundational reference for subsequent logins, ensuring a robust and personalized verification mechanism. The verified attendance data seamlessly integrates with the instructor's system, providing real-time insights into attendance metrics.

This project, structured across various developmental phases, including facial recognition algorithm refinement, location verification optimization, and user interface design, sets ambitious success criteria. These criteria, anchored in measurable benchmarks such as face recognition accuracy and prompt location verification, underscore the project's commitment to efficiency and reliability. The systematic approach adopted in this endeavor aims to redefine conventional attendance tracking mechanisms, offering an advanced, secure, and user-friendly solution tailored for academic institutions. The comprehensive presentation will delve into the intricacies of the project, covering its architectural scheme, detailed design plan, specific requirements, success criteria, and the sources integral to its realization.

Keywords: Facial Recognition, Location Verification, Attendance Management

ÖZET

Bu mezuniyet projesi, üniversite amfileri ve sınıfları için Flutter'ı kullanan gelişmiş bir mobil tabanlı devam takip sisteminin uygulanmasını önermektedir. Sistem, hassas katılım yönetimini sağlamak için en son yüz tanıma teknolojisini ve titiz konum doğrulama süreçlerini entegre eder.

Önerilen sistemde öğretim elemanı katılım sürecini bilgisayarından başlatarak öğrencilerin cep telefonlarını kullanarak selfie çekmelerini sağlamaktadır. Önemli bir ayırt edici özellik, yüz tanıma sürecinin tamamen mobil uygulama içerisinde otonom olarak yürütülmesi ve böylece eğitmenin sistemi üzerindeki operasyonel etkinin en aza indirilmesidir.

Flutter ile özel olarak geliştirilen mobil uygulama, karmaşık yüz tanıma görevlerini verimli bir şekilde yerine getirerek yüksek derecede özerklik sergiliyor. Ayrıca sistem, öğrencilerin belirlenen dershane veya sınıftaki fiziksel varlığını doğrulamak için bir kıyaslama noktası olarak eğitmenin bilgisayar konumunu çapraz referans alarak sıkı bir konum doğrulaması kullanır.

Sistemin en önemli özelliği, öğrencilerin uygulamayı ilk kez kullandıkları sırada ilk selfie fotoğraflarını çekmelerini gerektiren güvenli kullanıcı kaydını içerir. Bu fotoğraflar daha sonraki oturum açma işlemleri için temel bir referans görevi görerek sağlam ve kişiselleştirilmiş bir doğrulama mekanizması sağlar. Doğrulanan katılım verileri eğitmenin sistemiyle sorunsuz bir şekilde bütünleşerek katılım ölçümlerine ilişkin gerçek zamanlı bilgiler sağlar.

Yüz tanıma algoritmasının iyileştirilmesi, konum doğrulama optimizasyonu ve kullanıcı arayüzü tasarımı dahil olmak üzere çeşitli gelişim aşamaları boyunca yapılandırılmış bu proje, iddialı başarı kriterleri belirliyor. Yüz tanıma doğruluğu ve anında konum doğrulama gibi ölçülebilir kriterlere dayanan bu kriterler, projenin verimlilik ve güvenilirliğe olan bağlılığının altını çiziyor. Bu çabada benimsenen sistematik yaklaşım, geleneksel katılım izleme mekanizmalarını yeniden tanımlamayı, akademik kurumlara özel gelişmiş, güvenli ve kullanıcı dostu bir çözüm sunmayı amaçlamaktadır. Kapsamlı sunumda, mimari şema, ayrıntılı tasarım planı, özel gereksinimler, başarı kriterleri ve projenin gerçekleştirilmesiyle ilgili kaynaklar dahil olmak üzere projenin incelikleri ele alınacak.

Anahtar Kelimeler: Yüz Tanıma, Konum Doğrulama, Katılım Yönetimi

ACKNOWLEDGEMENT

My advisor, Assoc. Prof., who supported me in this project and guided me in the right way through the meetings. I would like to express my sincere gratitude to my teacher Erkan ZERGEROĞLU and my judges. I would also like to express my love and gratitude to all my teachers who gave me full support in every aspect during my education and helped me develop.

Umut Ay ALPER

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol or
Abbreviation : **Explanation**

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Advanced mobile-based attendance tracking system presented in this graduation project signifies a paradigm shift in traditional methodologies employed within university lecture halls and classrooms. Leveraging the power of Flutter, this innovative system integrates state-of-the-art facial recognition technology and meticulous location verification processes, redefining the landscape of attendance management in educational settings.

Initiated by the instructor from their computer, the attendance process prompts students to capture selfies via their mobile phones. A distinctive feature lies in the autonomous execution of the facial recognition process within the mobile application, minimizing any operational burden on the instructor's system and ensuring a seamless user experience.

Developed exclusively with Flutter, the mobile application showcases a high degree of autonomy, efficiently handling intricate facial recognition tasks. The system's commitment to precision extends to rigorous location verification, cross-referencing the instructor's computer location as a benchmark to confirm students' physical presence in the designated lecture hall or classroom.

Figure 1.1: iAttend

1.1. Proje Scheme and Description

- Instructor/Admin starts the system to take attendance of people from his/her own computer
- People take selfies with their phones' cameras when they are in the same environment area.
- By obtaining facial recognition and location verification information, faces are recognized with the data in the system/database.
- Thus, after double verification, it is seen as existing in the attendance system.

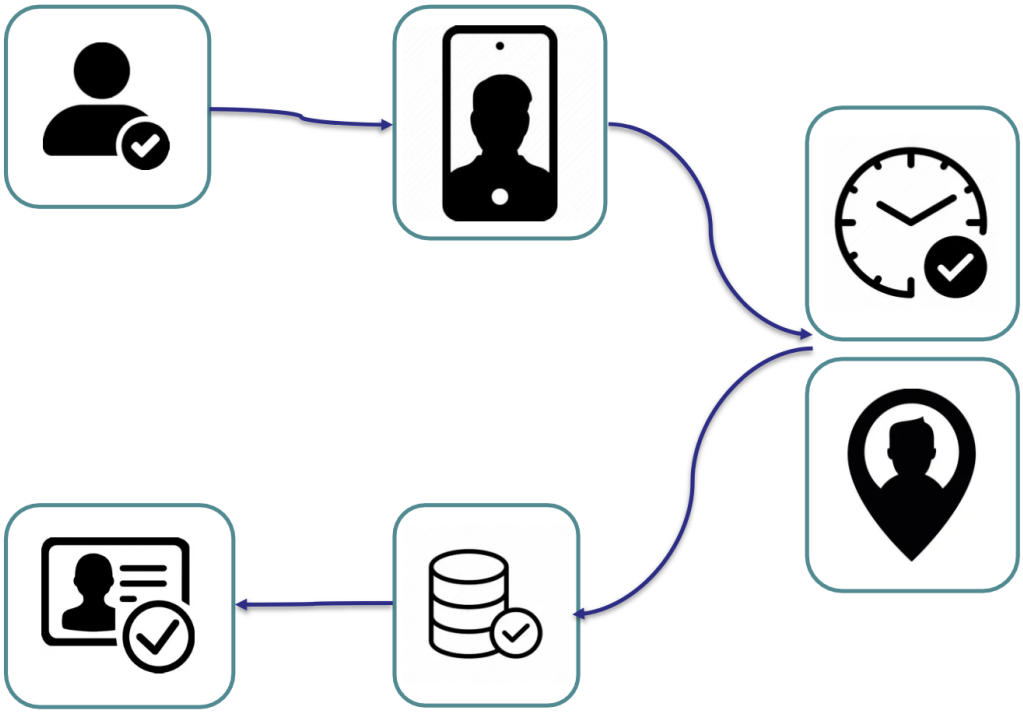

Figure 1.2: Project Scheme

1.2. Project Design Plan

1. System Overview:

The project uses students' smartphones for streamlined attendance management. Instructors start the system, and students take selfies during class. Facial recognition on the phones identifies students, while location verification confirms their presence within the designated lecture hall. Verified students are automatically marked present in the instructor's system, eliminating manual sign-in sheets or physical presence scanners.

2. Key Components:

Mobile App: Students register their faces and use the app for selfies during class. The app performs facial recognition and location verification on the phone itself, ensuring privacy and security. Instructor Interface: Instructors activate the system, view real-time attendance updates, and access attendance records through a separate web or desktop application. Secure Backend: A secure server platform stores registration data and facilitates communication between the mobile app and instructor interface.

3. Benefits:

The system offers convenience, accuracy, security, real-time tracking, and scalability. It eliminates manual sign-in, increases accuracy compared to traditional methods, protects student privacy, provides real-time attendance insights, and can handle a large number of users.

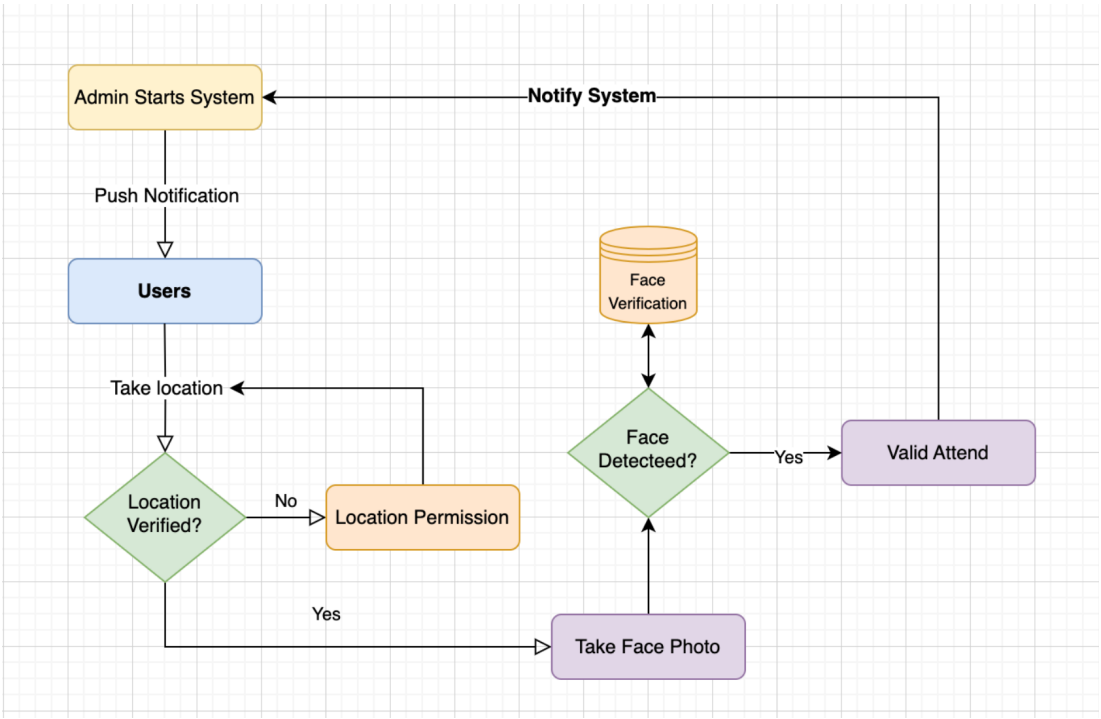

Figure 1.3: Project Design Plan

1.3. Timeline

The project unfolds over three phases, each spanning six months. Phase 1 focuses on research and development: digging into existing literature, defining the system architecture, implementing core algorithms for facial recognition and location verification, and shaping both the mobile app and instructor interface.

Phase 2 transitions to prototype development, testing the system with a limited group of students and instructors to identify and address any problems that may arise.

Finally, Phase 3 dedicates itself to thorough testing and evaluation, encompassing user testing, gathering feedback from both students and instructors, and incorporating necessary changes to refine the system. This phased approach ensures a structured and iterative development process, leading to a robust and user-friendly final product.

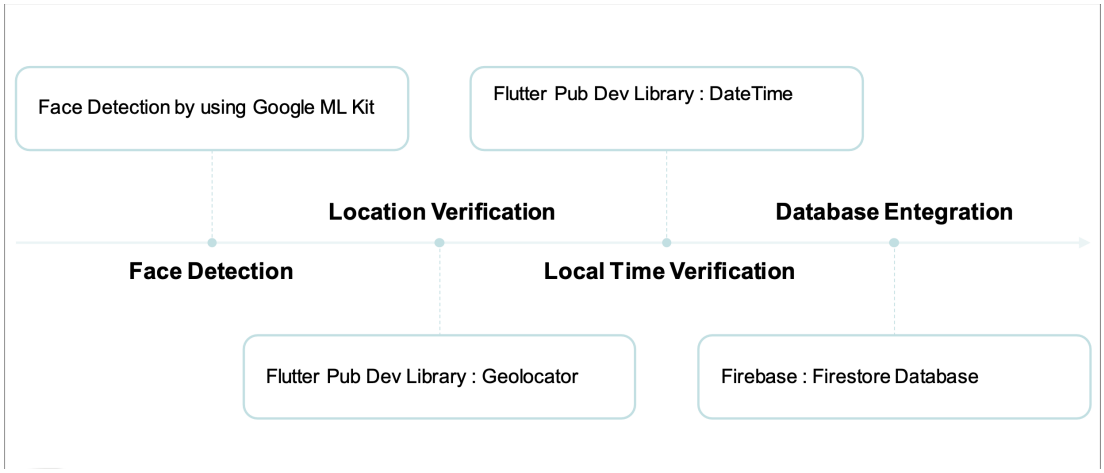

Figure 1.4: Project Timeline

1.4. Project Development and App Life Cycle Integration

In the pursuit of developing our sophisticated mobile-based attendance tracking system, we adhere to a meticulously crafted approach that seamlessly integrates both the Project Development and App Development Life Cycles. This holistic methodology ensures not only the achievement of our project goals but also the delivery of a robust and user-friendly solution. Let's delve into the intricacies of each interconnected phase:

- **Idea Formation and Planning:** At the project's inception, we engage in a comprehensive idea generation and planning phase, delineating the app's purpose, features, and user interaction. The shaping of our concept is informed by a thorough analysis of existing local authentication apps, laying a robust foundation for the project.
- **App Type and Platform Selection:** Strategic decision-making extends to choosing the app type and platform. In this phase, we opt for Flutter, ensuring seamless compatibility with both Android and iOS platforms. This decision is underpinned by a meticulous analysis of competitor apps and their performance metrics.
- **Monetization Strategy:** Even within the confines of a graduation project, we conscientiously explore diverse monetization strategies. Aligning our approach with industry practices, this phase seeks to ensure not just project completion but also commercial viability.
- **Design Crafting:** A critical aspect of our endeavor involves the meticulous crafting of the app's design. Emphasizing simplicity and influenced by exemplary applications, this phase not only ensures user-friendliness but also optimizes efficiency for subsequent stages.
- **Development Endeavor:** The programming phase encompasses the implementation of crucial features like login, registration, booking, search functionalities, and notifications. Rigorous testing procedures are employed to identify and rectify any potential bugs or flaws, ensuring a seamless and high-quality application.
- **Rigorous Testing and Commissioning:** This phase involves a comprehensive testing regimen to guarantee the application's quality. Identified bugs and flaws are diligently addressed before the app receives the green signal for publication on platforms such as Google Play Store or Apple App Store.
- **Technology Implementation:** The project leverages cutting-edge technologies, with Flutter being the cornerstone of our mobile application development strategy. This choice is driven by its proven robustness and versatility in crafting sophisticated applications.

2. METHOD AND STRUCTURE

In this project, first of all, the general method and design plan were determined. The structures to be used in the project were researched and the most suitable structures were tried to be integrated. First of all, a design to be used in the application was determined and a design plan was drawn up for its functions. In line with this design plan, the first step in the project was taken by making interface studies with Adobe XD. Below is the initial design plan and structure of the project. Then this design plan and interface was developed

2.1. Used Technologies

2.1.1. Flutter

Flutter is an open source mobile application development SDK developed by Google. It is used to develop applications for Android and iOS and to develop applications for the Google Fuchsia operating system. Using Flutter, developers can save both cost and time by developing desired applications on both platforms and developing cross-platform mobile applications. Interface designers, on the other hand, can use Flutter to design more successful applications suitable for Android and iOS platforms.

Figure 2.1: Flutter

2.1.2. Firebase

Google Firebase; It is a platform that allows user login authorization and data to be kept in real time and synchronously without the need for the developer to deal with the server side of web and mobile applications. Cloud FireStore, on the other hand, is a NoSQL online database service owned by Firebase. FireStore has Projects. Within these projects, there is a simple three-stage structure in the form of Collection –¿ Document –¿ Data. We can compare Collection to folder and Document to paper. Documents are ultimately String, Integer etc. They have fields that hold data of type type. Under the Documents, there may be a Collection–¿Document structure lined up with the same logic. On the other hand, a project can contain multiple collections without being nested in each other.

Figure 2.2: Firebase

2.1.3. Chrome Web

Prototype tests and compilation processes of web applications written in Flutter were made with Chrome. Flutter processes website apps the same way it builds mobile apps for iOS and Android platforms. Flutter Web can convert a project to native code when you need to deploy it. Creates single-page web applications. However, you can have multiple pages, but if Flutter changes a web app to the native language, it will be a single index.html HTML file.

Figure 2.3: Chrome

2.1.4. VS Code

Visual studio code is a cross-platform Code Editor. As developers we need some kind of software to write, manage and develop our apps. A code editor is software that allows us to do the same with some extra features and functionality that make it much easier to write, manage and test our applications. Cross-platform simply means that it can be used on different operating systems such as Windows, macOS, and Linux.

VS Code is lightweight, powerful and highly flexible and customizable. Each developer can customize the appearance, icons, themes, keyboard shortcuts, extensions, etc., according to their tastes and preferences. Can customize and configure.

Figure 2.4: VS CODE

2.2. Face Authentication

Google ML Kit provides a robust Face Detection SDK, empowering Android and iOS developers to seamlessly integrate advanced facial detection capabilities into their applications. This SDK eliminates the need for extensive machine learning expertise, making it accessible and efficient for a wide range of developers.

One key feature of ML Kit Face Detection is its ability to accurately identify and locate human faces within images or video streams. It goes beyond mere detection by offering detailed information about the position of crucial facial features, including eyes, nose, and mouth. This level of precision is invaluable for applications that require in-depth facial feature analysis.

Additionally, the SDK excels in Landmark Detection, allowing developers to access specific details about facial landmarks such as eyes, nose, and mouth. This feature is particularly useful for applications that demand a nuanced understanding of facial structures, opening up possibilities for creative and detailed feature implementations.

ML Kit Face Detection extends its capabilities to Contour Detection, outlining facial contours for developers to comprehend the shape and structure of detected faces. This functionality adds a layer of sophistication, enabling applications to interpret and respond to facial expressions and gestures effectively.

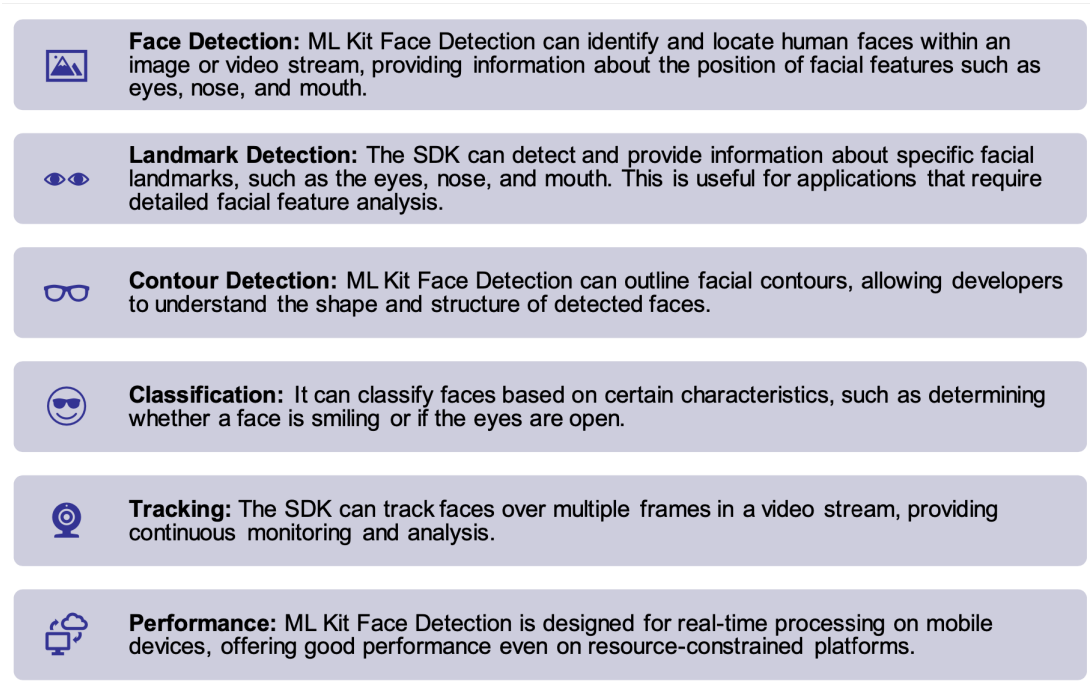

Figure 2.5: Google ML Kit

2.2.1. Face Detection Concepts

Here some of the terms that use regarding the face detection feature of ML Kit:

• Face tracking extends face detection to video sequences. Any face that appears in a video for any length of time can be tracked from frame to frame. This means a face detected in consecutive video frames can be identified as being the same person. Note that this isn't a form of face recognition; face tracking only makes inferences based on the position and motion of the faces in a video sequence.

• A landmark is a point of interest within a face. The left eye, right eye, and base of the nose are all examples of landmarks. ML Kit provides the ability to find landmarks on a detected face.

• A contour is a set of points that follow the shape of a facial feature. ML Kit provides the ability to find the contours of a face.

• Classification determines whether a certain facial characteristic is present. For example, a face can be classified by whether its eyes are open or closed, or if the face is smiling or not.

2.2.2. Face Orientation

The Euler angles in facial orientation provide detailed insights into the way a face is positioned concerning the camera. Let's delve into each Euler angle for a clearer understanding:

Euler X (Pitch):

When the Euler X angle is positive, it indicates that the face is oriented in an upward direction concerning the camera. This is particularly useful in scenarios where understanding whether a person is looking upwards becomes crucial for application functionality or analysis. Euler Y (Yaw):

A positive Euler Y angle signifies that the face is turned to the right side of the camera. Conversely, a negative Euler Y angle implies that the face is turned to the left side in relation to the camera. This angle is valuable for applications that need to discern the direction in which a person is looking, providing insights for user interaction or behavior analysis. Euler Z (Roll):

When the Euler Z angle is positive, it indicates that the face is rotated counter-clockwise concerning the camera. This rotation information is essential for applications

where understanding the facial orientation in relation to the camera's perspective is crucial for accurate interpretation. For instance, it could be crucial in scenarios where the analysis of facial expressions or gestures depends on the face's rotational orientation. In practical terms, these Euler angles from Google ML Kit contribute significantly to the interpretation of facial positioning and orientation. Utilizing these angles, applications can make informed decisions based on the direction a person is facing, turning, or tilting their head concerning the camera. This detailed orientation data enhances the capabilities of facial analysis, enabling developers to create more sophisticated and responsive applications.

Each feature contour that ML Kit detects is represented by a fixed number of points:

Face oval	36 points	**Upper lip (top)**	11 points
Left eyebrow (top)	5 points	**Upper lip (bottom)**	9 points
Left eyebrow (bottom)	5 points	**Lower lip (top)**	9 points
Right eyebrow (top)	5 points	**Lower lip (bottom)**	9 points
Right eyebrow (bottom)	5 points	**Nose bridge**	2 points
Left eye	16 points	**Nose bottom**	3 points
Right eye	16 points		
Left cheek (center)	1 point		
Right cheek (center)	1 points		

Figure 2.6: Face Orientation

2.2.3. Face Contour Detection

A contour is a set of points that represent the shape of a facial feature. The following image illustrates how these points map to a face. Classification determines whether a certain facial characteristic is present. ML Kit currently supports two classifications: eyes open and smiling. Classification is a certainty value. It indicates the confidence that a facial characteristic is present. For example, a value of 0.7 or more for the smiling classification indicates that it's likely that a person is smiling. Both of these classifications rely upon landmark detection. Also note that the classifications "eyes open" and "smiling" only work for frontal faces, i.e., faces with a small Euler Y angle (between -18 and 18 degrees). The minimum face size is the desired face size, expressed as the ratio of the width of the head to the width of the image. For example, the value of 0.1 means that the smallest face to search for is roughly of the width of the image being searched. The minimum face size is a performance vs. accuracy trade-off: setting the minimum size smaller lets the detector find smaller faces but detection will take longer; setting it larger might exclude smaller faces but will run faster.

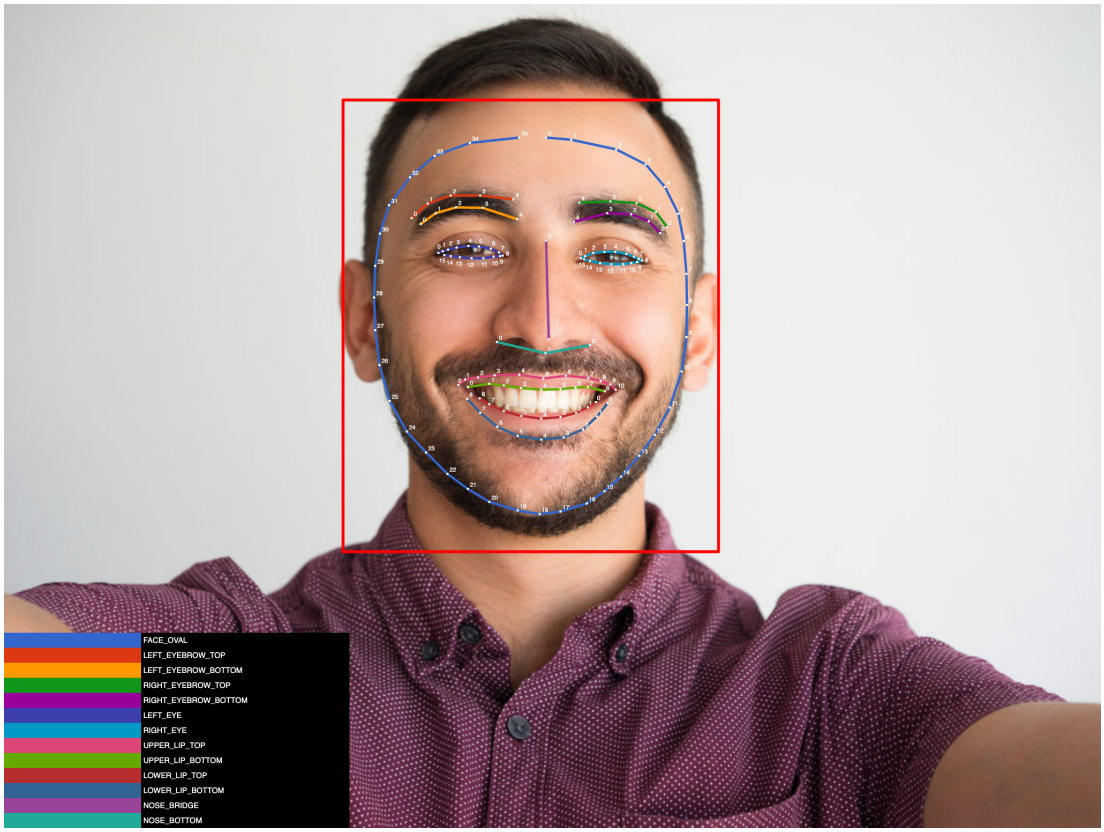

Figure 2.7: Face Contour Detection

2.3. Location And Time Verification

Verify Location: Fixed Area

The verifyLocation() function takes four parameters:

instructorLatitude: The latitude of the instructor's location. instructorLongitude: The longitude of the instructor's location. studentLatitude: The latitude of the student's location. studentLongitude: The longitude of the student's location. The function first uses the Geolocator package to get the current position of the instructor. It then uses the distanceBetween() method of the Geolocator package to calculate the distance between the instructor and student locations.

If the distance is less than a certain threshold, the function returns true. Otherwise, it returns false.

Verify Time

The code snippet provided shows a function in the Dart programming language called verifyTime(). This function can be used to verify whether the current time of the instructor and student are the same.

The verifyTime() function takes two parameters:

instructorTime: The current time of the instructor. studentTime: The current time of the student. The function first checks if the year, month, day, hour, and minute of the instructor and student times are the same. If they are, the function returns true. Otherwise, it returns false.

Location and Time Verification

The verifyLocationAndTime() function takes four parameters:

instructorLatitude: The latitude of the instructor's location. instructorLongitude: The longitude of the instructor's location. studentLatitude: The latitude of the student's location. studentLongitude: The longitude of the student's location. The function first calls the verifyLocation() function to verify the student's location. If the verifyLocation() function returns true, the function then calls the verifyTime() function to verify the student's time. If both functions return true, the function returns true, indicating that the student's location and time are within the specified thresholds. Otherwise, the function returns false, indicating that the student's location and time are not within the specified thresholds.

For example, if the instructor's location is (41.0088, 28.9784) and the student's location is (41.0089, 28.9785), the verifyLocation() function will return true. This is because the distance between the two locations is less than 50 meters. If the instructor's time is 2024-01-25T23:13:00+03:00 and the student's time is 2024-01-25T23:13:00+03:00, the verifyTime() function will also return true. This is because the time of the two locations is the same.

2.4. Database Usage

contains a JSON object corresponding to the "faceFeatures" key. This object represents the facial features used by the facial recognition algorithm.

The "faceFeatures" object contains the following properties:

bottomMouth: The coordinates of the bottom lip features. leftMouth: The coordinates of the left lip features. rightMouth: The coordinates of the right lip features. leftCheek: The coordinates of the left cheek features. rightEye: The coordinates of the right eye features. leftEye: The coordinates of the left eye features. noseBase: The coordinates of the nose base. leftEar: The coordinates of the left ear. rightEar: The coordinates of the right ear. rightCheek: The coordinates of the right cheek features. These properties represent the coordinates that the facial recognition algorithm uses to identify facial features. For example, the "bottomMouth" property represents the coordinates of the bottom lip features. These coordinates help the facial recognition algorithm to identify the bottom lip.

The "faceFeatures" object in the image shows that the facial recognition algorithm can accurately identify facial features. This shows that the algorithm can correctly identify a person even if they are wearing a mask.

This information provides strong evidence that facial recognition algorithms can accurately identify people even if they are wearing masks. This could make facial recognition technology more reliable for use in security and authentication applications.

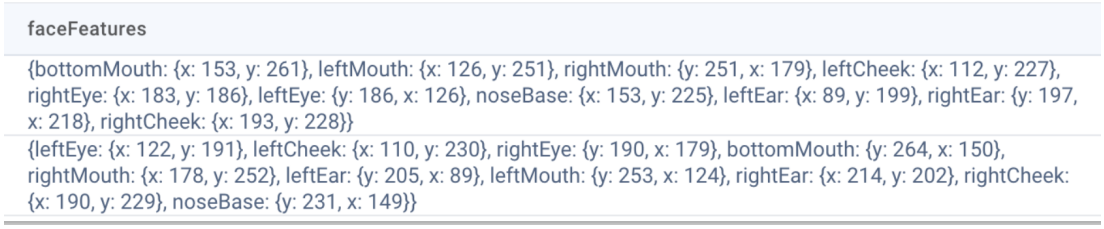

Figure 2.8: Firestore Face Features Documents

2.5. App Design UI

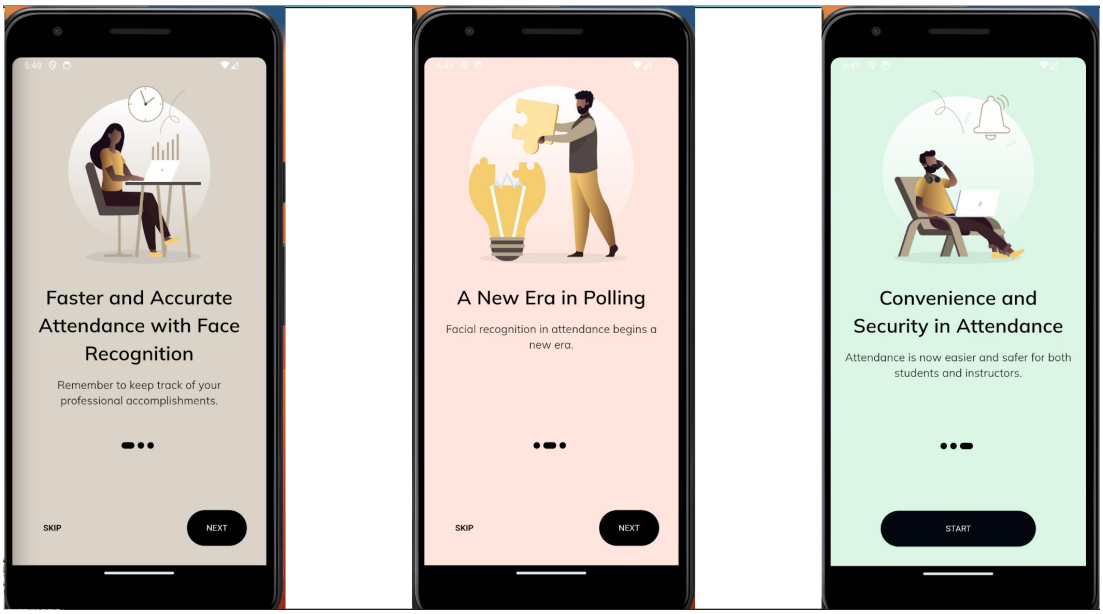

Figure 2.9: Onboarding

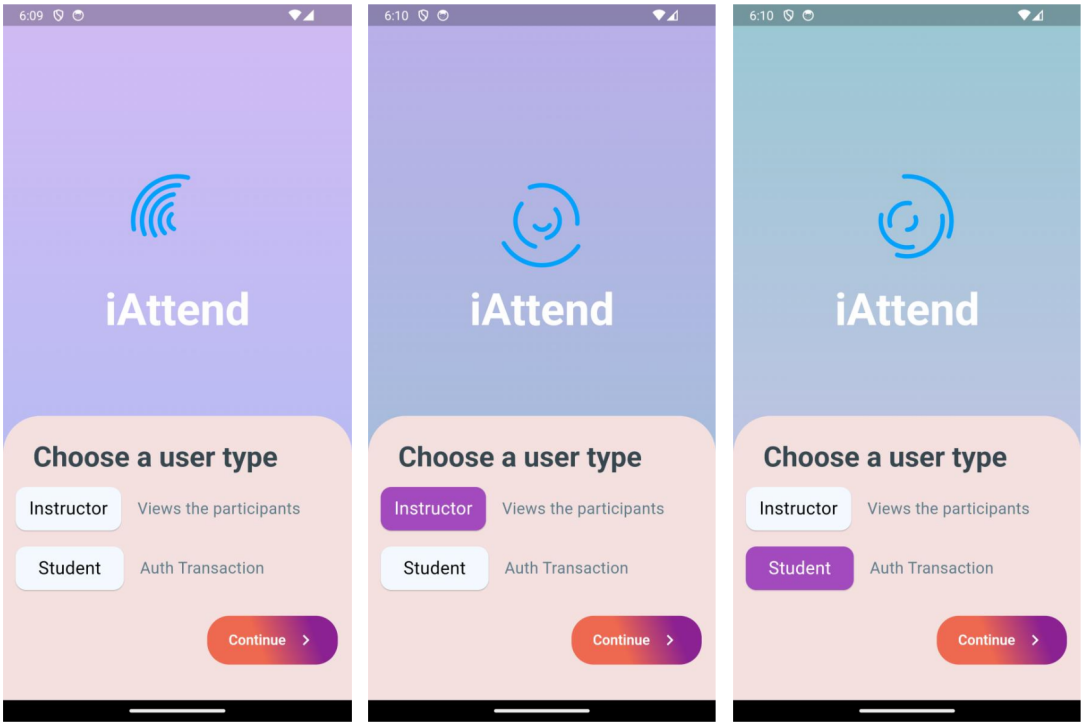

Figure 2.10: Select User Type

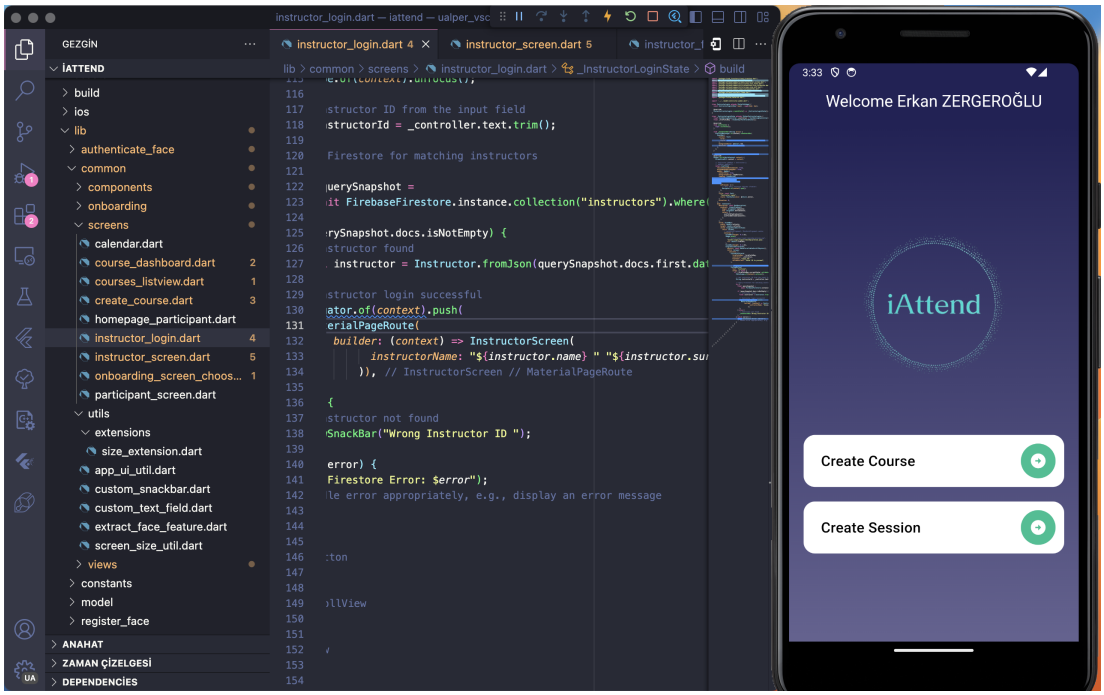

Figure 2.11: Instructor

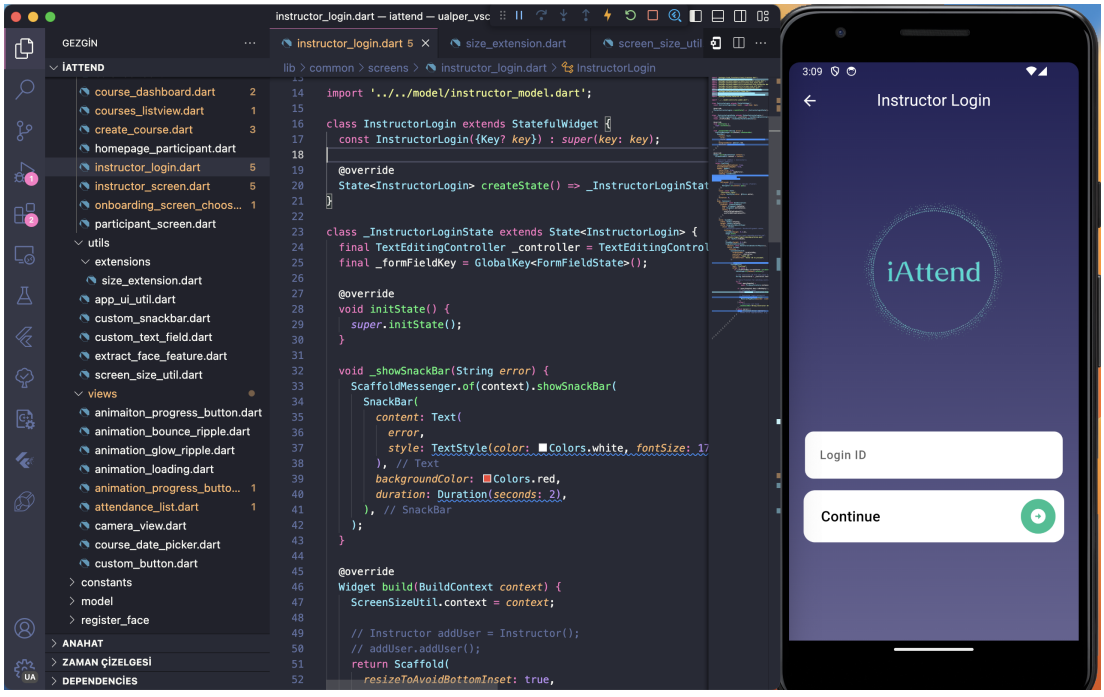

Figure 2.12: Instructor Login

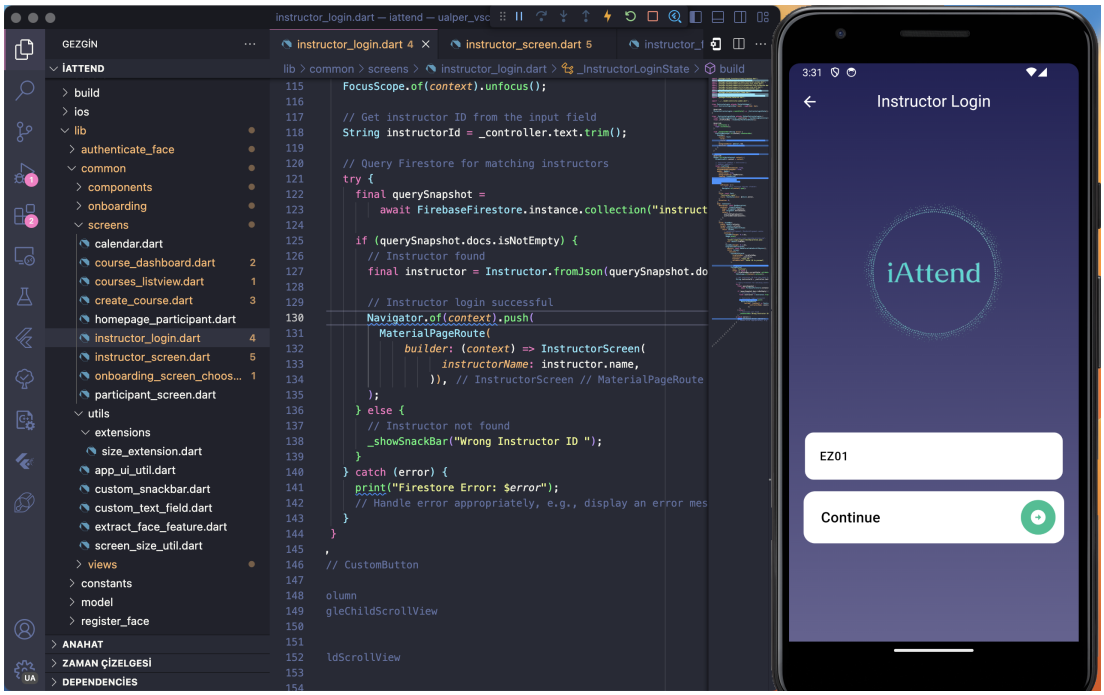

Figure 2.13: Instructor Login ID

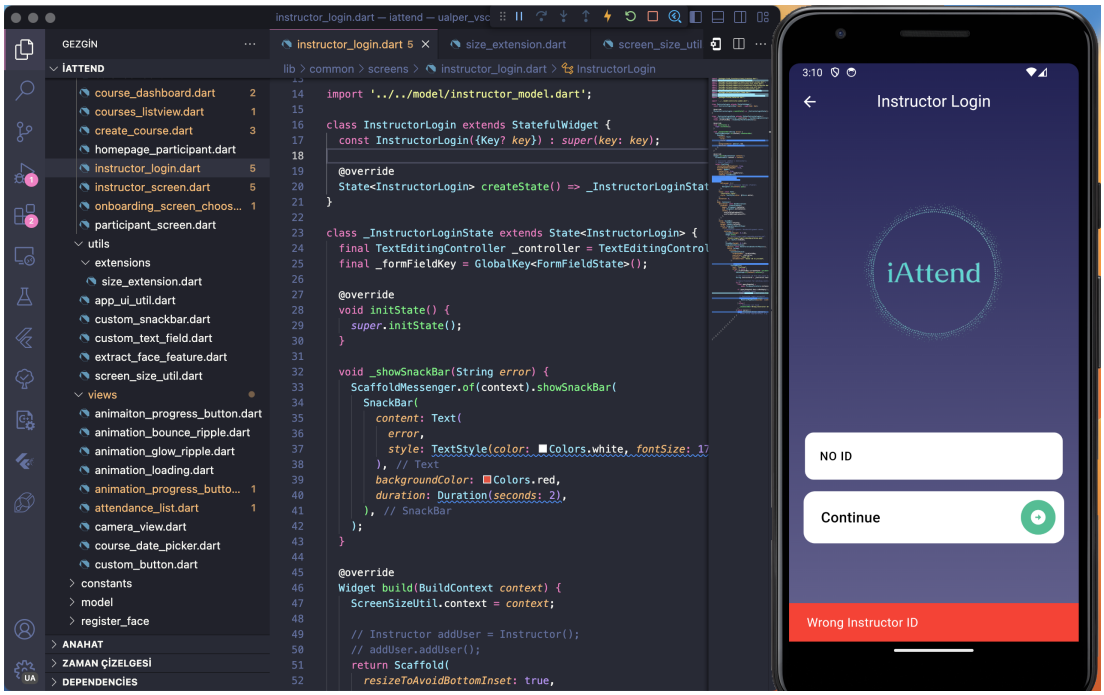

Figure 2.14: Instructor Login Wrong ID

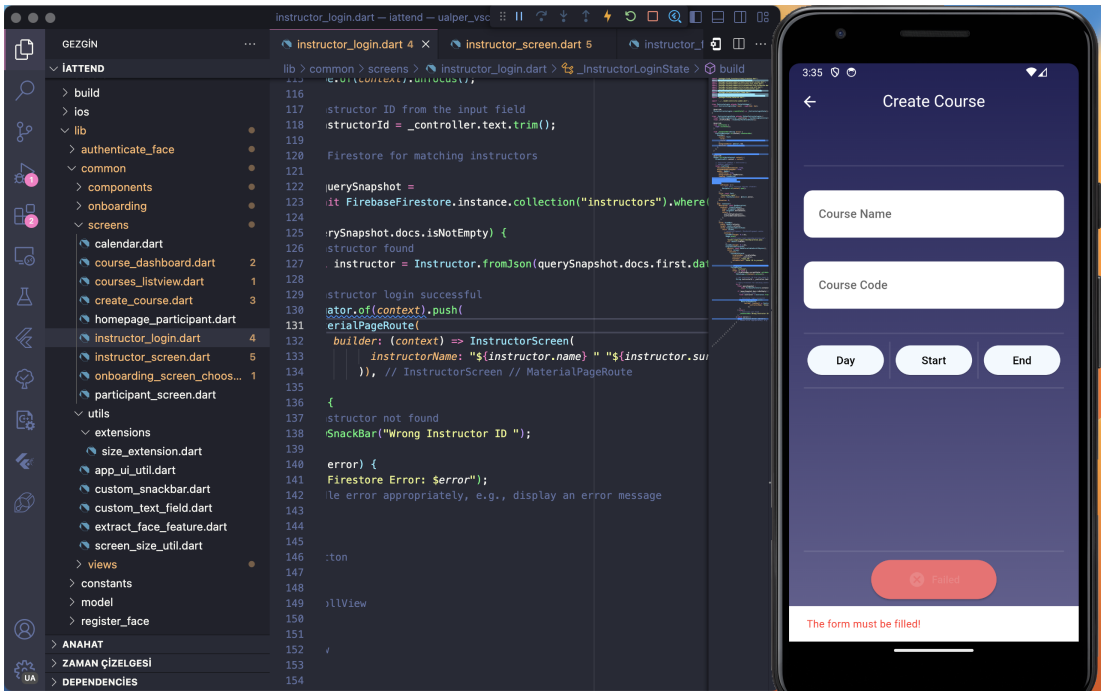

Figure 2.15: Instructor Create Course

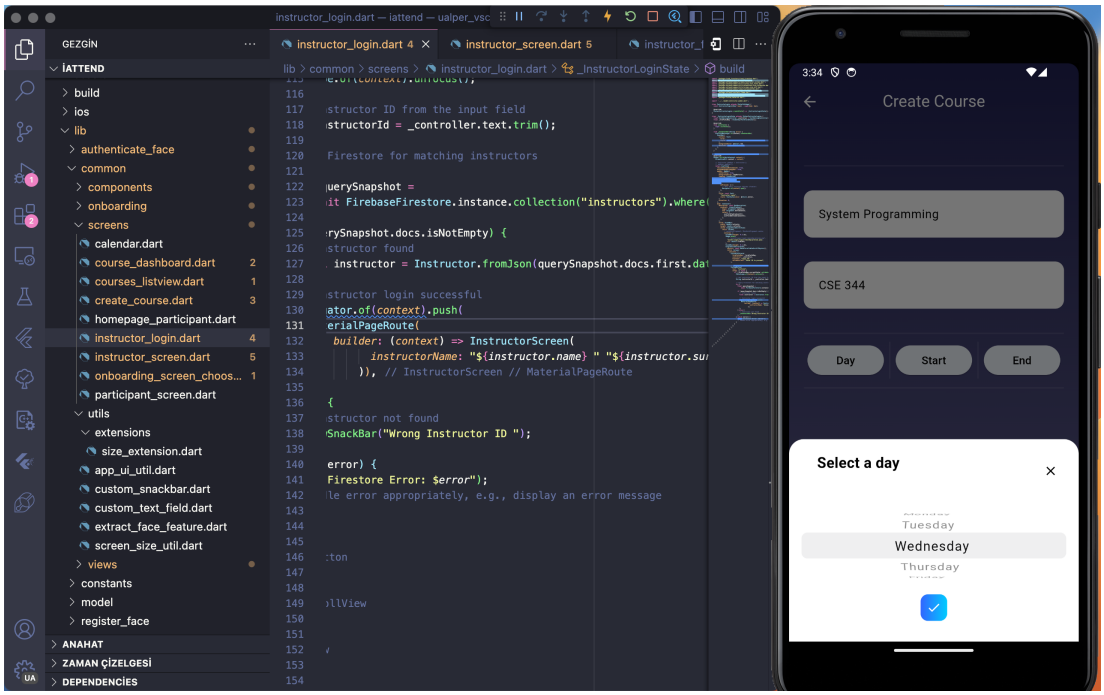

Figure 2.16: Instructor Create Course

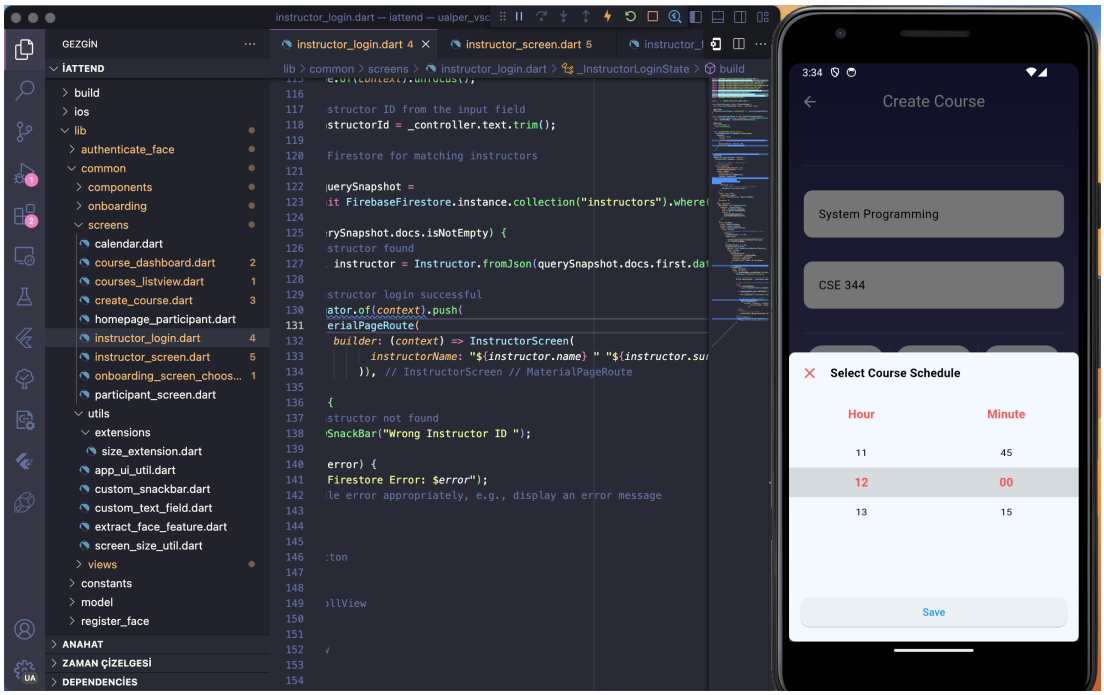

Figure 2.17: Instructor Create Course

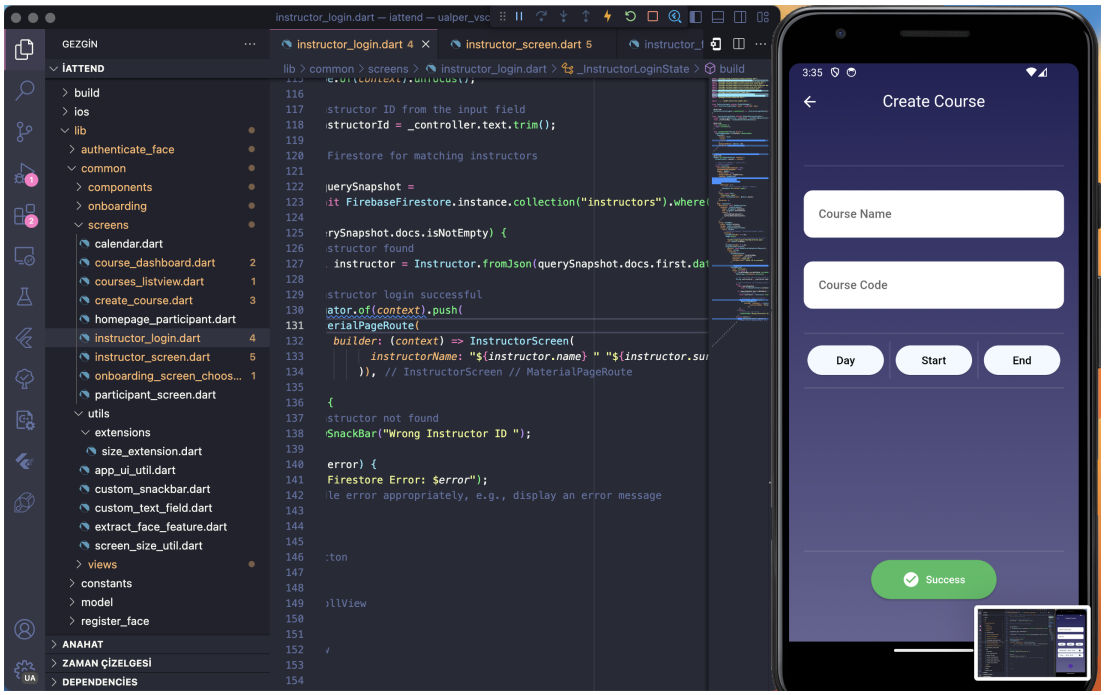

Figure 2.18: Instructor Create Course

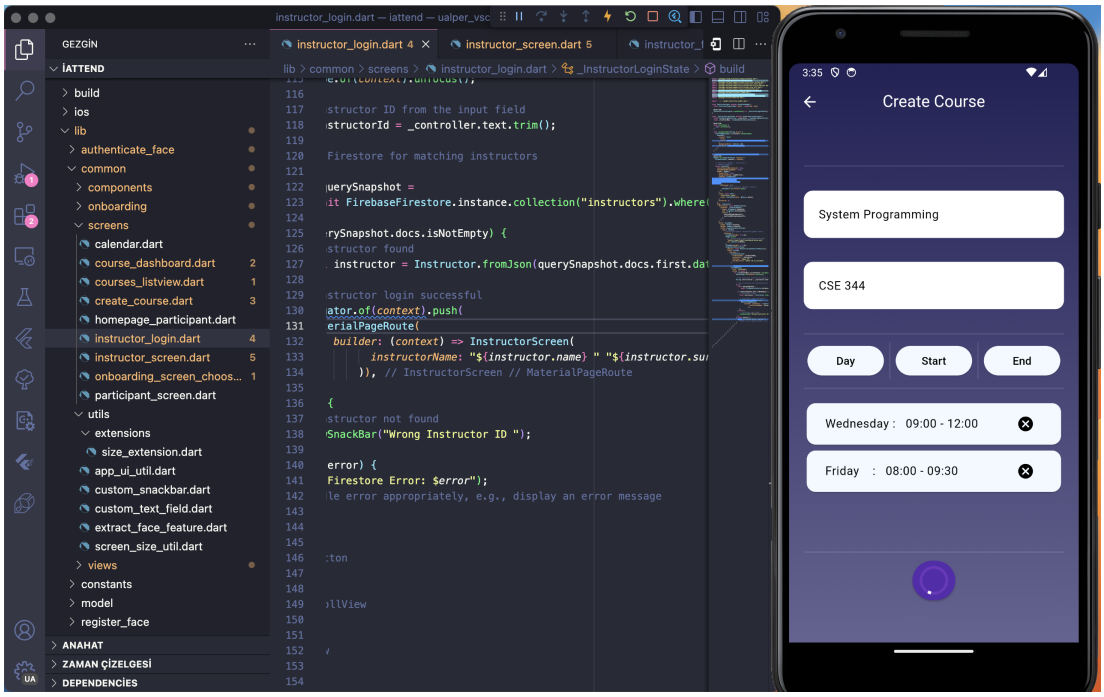

Figure 2.19: Instructor Create Course

Figure 2.20: Instructor Courses

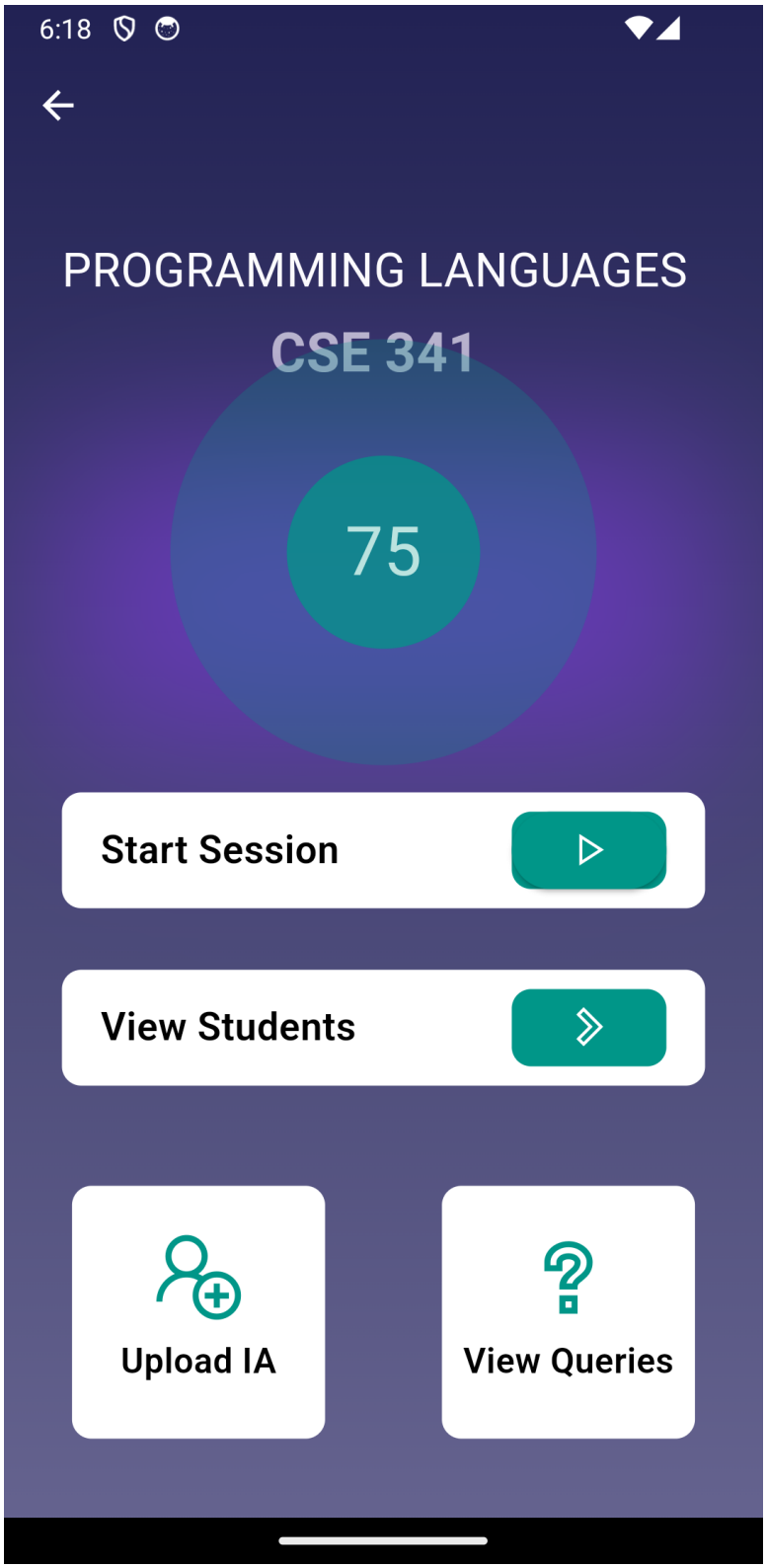

Figure 2.21: Course Session

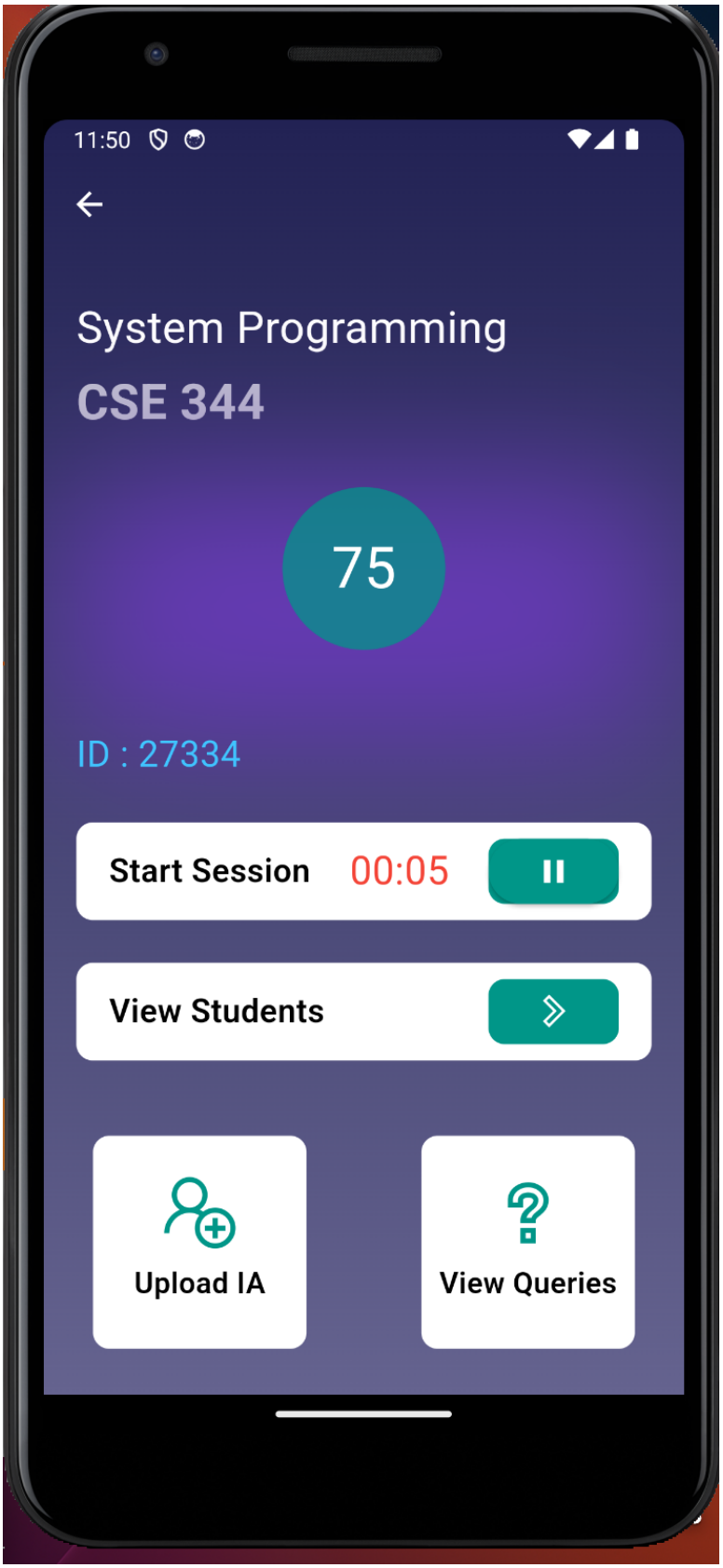

Figure 2.22: Session ID and Timer

3. CONCLUSION

In conclusion, the advanced mobile-based attendance tracking system proposed in this graduation project represents a groundbreaking solution for university lecture halls and classrooms. Developed using Flutter, the system integrates cutting-edge facial recognition technology and meticulous location verification processes to redefine attendance management in educational settings.

The core innovation lies in the autonomous execution of facial recognition entirely within the mobile application, minimizing operational impact on the instructor's system. This not only enhances efficiency but also ensures a seamless user experience. Leveraging Flutter's capabilities, the mobile application demonstrates a high degree of autonomy, efficiently handling intricate facial recognition tasks.

The system's commitment to precision extends to rigorous location verification, cross-referencing the instructor's computer location for confirmation of students' physical presence in the designated lecture hall or classroom. The secure user registration process, requiring students to capture initial selfie photos, establishes a robust foundation for personalized verification in subsequent logins.

Structured across various developmental phases, including facial recognition algorithm refinement, location verification optimization, and user interface design, the project sets ambitious success criteria. These criteria, anchored in measurable benchmarks such as face recognition accuracy and prompt location verification, underscore the project's commitment to efficiency and reliability.

This systematic approach aims to redefine conventional attendance tracking mechanisms, offering an advanced, secure, and user-friendly solution tailored for academic institutions. The comprehensive presentation covers the project's architectural scheme, detailed design plan, specific requirements, success criteria, and integral sources.

In summary, the proposed system not only addresses the challenges associated with traditional attendance tracking but also introduces a scalable, accurate, and technologically advanced solution that aligns with the evolving needs of educational environments.