NJALA UNIVERSITY

SCHOOL OF POSTGRADUATE STUDIES
DEPARTMENT OF COMPUTER SCIENCE

BIOMETRIC-AUTHENTICATED MOBILE INVOICING SYSTEM

A Semester 3 Project Submitted in Partial Fulfillment of the Requirements for the Award of Master of Science in Computer Science

Candidates Name: Abu Junior Vandi and Alie Abubakarr
Student ID: *82937 & 16873*
Supervisor: Mr. A.J. Fofanah
Course Code: CSD634 – Mobile Software Development
Semester: Third Semester
Submission Date: July 15, 2025

Abstract

In the digital era, managing financial transactions securely and efficiently remains a major concern, especially in small businesses and freelance environments. This project presents the development of a Biometric-Authenticated Mobile Invoicing System, an all-in-one React Native application that enables users to generate, manage, and send invoices directly from their smartphones while securely authenticating users using biometric data such as fingerprints or facial recognition.

The app integrates two core systems: a mobile invoicing system built with React Native and JavaScript, and a biometric authentication system developed in Python. Together, these modules provide a seamless user experience with robust security for sensitive financial operations. Key features include invoice generation, client management, payment tracking, secure login with biometrics, and local/remote data storage.

The project showcases modern mobile development principles, cross-platform UI design using Figma, secure biometric data handling, and the use of APIs and local storage. The system aims to enhance productivity and improve data protection for financial apps in finance-related businesses and professionals.

Table of Contents

BIOMETRIC-AUTHENTICATED MOBILE INVOICING SYSTEM..... 1
List of Abbreviations..... 3
Introduction..... 3
1.1 Background..... 3
1.2 Problem Statement..... 4
1.3 Objectives..... 4
General Objective:..... 4
Specific Objectives:..... 4
1.4 Scope of the Study..... 4
1.5 Significance of the Study..... 4
1.6 Methodology..... 5
Literature Review..... 5
2.1 Introduction..... 5
2.2 Mobile Invoicing Systems..... 5
2.3 Biometric Authentication in Mobile Applications..... 6
2.4 Integration of Biometrics with Financial Systems..... 6
2.5 Cross-Platform Mobile Development with React Native..... 6
2.6 Python for Biometric Systems..... 6
2.7 Gaps in Existing Systems..... 7
2.8 Summary..... 7
System Analysis and Design..... 7
3.1 Architecture Diagram..... 7
3.2 Data Flow Diagram (DFD)..... 9
3.3 UI/UX Design..... 9
3.4 Security Measures..... 11
Implementation and Development..... 11
4.1 Tools and Technologies..... 11
4.2 Key Features..... 12
4.3 System Components..... 12
4.4 API Integration..... 13
4.5 Data Storage and Authentication..... 14
Testing and Evaluation..... 15
5.1 Unit Testing..... 15
5.2 Integration Testing..... 16
5.3 User Testing..... 17
Conclusion and Recommendations..... 18
6.1 Conclusion..... 18
6.2 Recommendations..... 18
References..... 19

List of Abbreviations

Abbreviation	Meaning
UI	User Interface
UX	User Experience
API	Application Programming Interface
DB	Database
JSON	JavaScript Object Notation
OTP	One-Time Password
MFA	Multi-Factor Authentication
RN	React Native
JS	JavaScript
DFD	Data Flow Diagram
IDE	Integrated Development Environment
OS	Operating System
SQL	Structured Query Language
ID	Identifier

Chapter 1

Introduction

1.1 Background

In today's fast-paced digital economy, small businesses, freelancers, and professionals are increasingly seeking convenient ways to manage their finances on the go. Invoicing is a core component of business operations, yet many traditional systems are desktop-bound, insecure, or overly complex. Meanwhile, the need for secure authentication methods has grown significantly, particularly for apps dealing with financial data.

This project proposes the development of a Biometric-Authenticated Mobile Invoicing System, a cross-platform mobile application that combines secure biometric login functionality with an intuitive invoicing interface. The solution leverages React Native for building the mobile frontend, and Python for handling biometric authentication. This integration provides a secure and user-friendly way for users to create, send, and track invoices from their smartphones, while ensuring that access to the app is tightly secured through fingerprint or facial recognition.

1.2 Problem Statement

Most mobile invoicing applications rely solely on PINs or passwords, which are often forgotten or compromised. Additionally, users need a fast, secure, and efficient system that simplifies invoice generation without sacrificing data security. There is a need for a lightweight, secure mobile invoicing solution that integrates biometric authentication, especially in environments where business is conducted remotely or across devices.

1.3 Objectives

General Objective:

To develop a mobile invoicing application integrated with a biometric authentication system to ensure security and enhance user experience.

Specific Objectives:

- To design and develop a React Native-based invoicing application.
- To implement biometric authentication (fingerprint/face ID) using a Python-based biometric system.
- To allow users to generate, store, and manage invoices on their mobile devices.
- To provide a modern, clean, and responsive user interface.
- To ensure secure data storage and encrypted biometric authentication.
- To test the application thoroughly for reliability, performance, and security.

1.4 Scope of the Study

This project focuses on two integrated components:

1. **Mobile Invoicing App** (React Native + JavaScript): Provides core features such as user registration/login, client management, invoice generation (payment tracking) and receipt generation.

2. **Biometric Authentication System** (Python): Captures, encrypts, stores, and verifies biometric data for user authentication locally.

The app is designed for Android and iOS devices and targets small business owners, freelancers, and professionals who require secure, mobile financial tools. The system does not include advanced accounting features like tax calculations or inventory management.

1.5 Significance of the Study

This study demonstrates how biometric security can be applied to finance-related mobile applications, addressing rising concerns about data privacy and authentication security. It provides a reference for integrating React Native applications with Python-based backend systems, contributing to both academic research and practical app development. The final product can serve as a prototype for fintech solutions in developing markets, where smartphones are the primary computing platform.

1.6 Methodology

The project was executed through the following phases:

1. **Requirement Gathering** – Identifying the user needs and functional goals.

2. **UI/UX Design** – Designing user interface screens in Canva with usability in mind.

3. **Frontend Development** – Developing the mobile app using React Native,Expo Go, and JavaScript.

4. **Backend Integration** – Implementing the biometric authentication using Python and connecting it to the mobile frontend.

5. **Testing** – Conducting unit, integration, and security testing.

6. **Documentation** – Compiling technical documentation, user manual, and evaluation results.

Chapter 2

Literature Review

2.1 Introduction

A literature review is essential to understand existing technologies, research trends, and practical solutions in mobile invoicing systems and biometric authentication. This chapter explores previous studies and systems in the domains of financial mobile applications, biometric security, and cross-platform mobile development using React Native and Python integration. The goal is to establish a foundation upon which the current project is built.

2.2 Mobile Invoicing Systems

Mobile invoicing apps have become essential tools for businesses and freelancers who require remote billing capabilities. Popular apps like Zoho Invoice, QuickBooks, and Invoice2go allow users to generate and send invoices via mobile devices. However, many of these applications rely on traditional password-based authentication, which is vulnerable to breaches.

Chukwu et al. (2021) found that mobile invoicing enhances cash flow and improves record-keeping, but stressed that security concerns (especially around client data and financial records) remain a significant barrier to wider adoption.

2.3 Biometric Authentication in Mobile Applications

Biometric authentication uses unique physical characteristics—like fingerprints or facial features—to confirm a user's identity. With the advent of smartphones equipped with biometric sensors, biometrics has become a preferred method for secure and seamless authentication.

According to Jain et al. (2016), biometric authentication offers a higher level of security compared to traditional credentials (PINs or passwords), especially when biometric data is encrypted and stored securely. In mobile contexts, biometric login not only improves security but also user convenience.

Recent studies, such as those by Oyediran and Bashir (2022), support the use of biometrics in fintech and e-commerce apps, citing enhanced fraud prevention and improved user experience.

2.4 Integration of Biometrics with Financial Systems

Combining financial systems with biometric authentication is gaining traction. Some banking apps now allow fingerprint or facial recognition for login and transaction confirmation.

However, the integration of custom biometric modules (e.g., built in Python) with mobile frontends is still relatively novel in academic research.

Projects such as FinBioPay (2020) demonstrated that small-scale biometric systems could be successfully integrated with mobile payment applications to enhance transaction-level security.

This project builds upon such ideas by integrating a Python-based biometric backend with a React Native frontend—a relatively underexplored area in academic mobile app development.

2.5 Cross-Platform Mobile Development with React Native

React Native, developed by Meta (formerly Facebook), is a popular framework for building cross-platform mobile applications using JavaScript. It allows code sharing between Android and iOS, reducing development time and ensuring UI consistency.

Studies by Kiran et al. (2019) concluded that React Native is efficient for small to mid-scale applications that require fast prototyping and frequent UI updates. Its ability to integrate with native modules and backend APIs makes it a solid choice for apps like the one proposed in this project.

2.6 Python for Biometric Systems

Python is a widely used language in machine learning and biometric systems due to its simplicity and vast library support. Libraries such as OpenCV, face_recognition, and Fingerprint SDKs allow developers to build facial and fingerprint recognition systems with relatively little overhead.

In this project, Python handles biometric data capture, encryption, and verification, which is then interfaced with the React Native app through a RESTful API.

2.7 Gaps in Existing Systems

From the reviewed literature and systems, the following gaps are identified:

- Most mobile invoicing apps do not implement biometric authentication.
- Custom biometric modules (e.g., fingerprint matching via Python) are rarely integrated into mobile React Native apps.
- Limited literature combines secure biometric login with user-generated financial content such as invoices.

2.8 Summary

This review has covered the key concepts and tools related to mobile invoicing systems, biometric authentication, and cross-platform mobile development. It highlights the potential

of integrating a Python-based biometric backend with a React Native mobile app to provide a secure, efficient, and user-friendly invoicing platform.

The project contributes to filling existing gaps by merging financial management with advanced biometric security in a mobile-first environment.

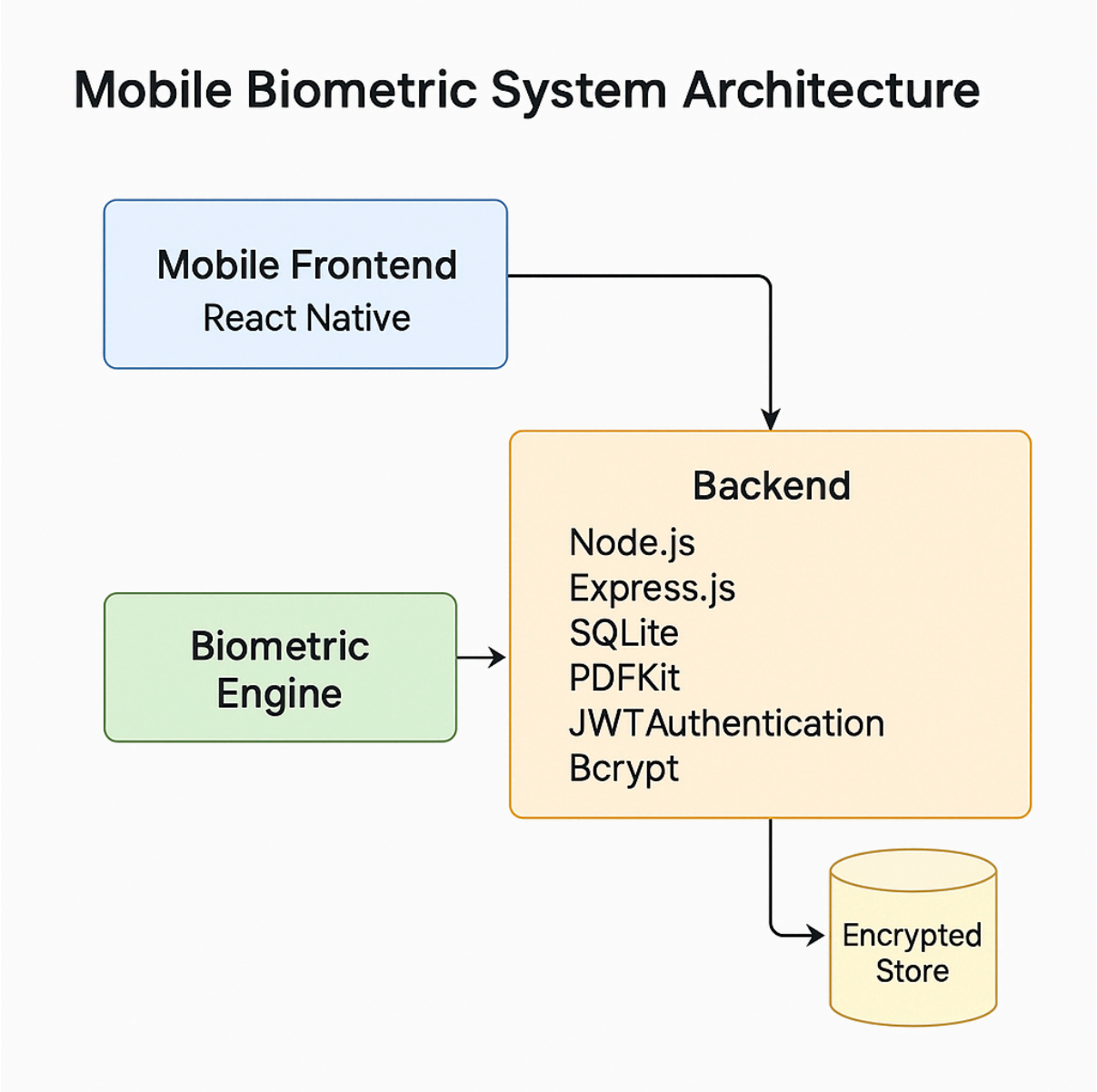
Chapter 3

System Analysis and Design

3.1 Architecture Diagram

The architecture of the application follows a modular client-server model that separates the mobile frontend (built with React Native) from the backend biometric system (developed in Python). This separation ensures maintainability, scalability, and secure data exchange.

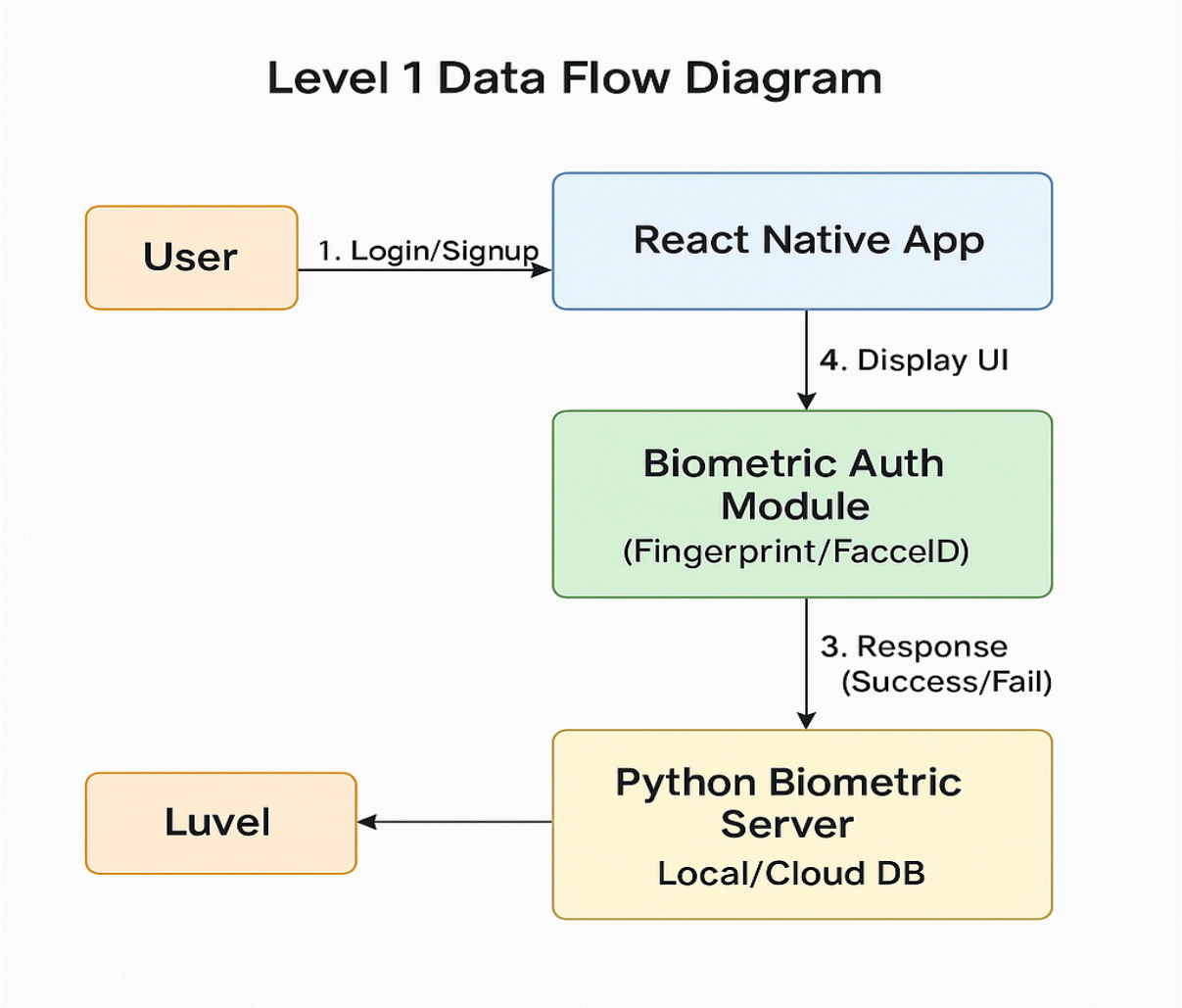
Fig 3.1: System Architecture Diagram



- **Frontend:** React Native mobile app handles user registration, login, invoice generation, and local display.
- **Backend API:** A lightweight Node.js and Express.js-based REST API serves as the middleware between the mobile app and the biometric engine, utilizing SQLite for data storage, PDFKit for document generation, JWT for authentication, and Bcrypt for password hashing.
- **Biometric Engine:** Built in Python using libraries like `face_recognition`, `OpenCV`, or `Fingerprint SDK`, responsible for capturing, storing, and matching biometric data.
- **Data Storage:** Sensitive data is encrypted and either stored locally (e.g., SQLite or AsyncStorage) or sent to a secure remote server.

3.2 Data Flow Diagram (DFD)

Fig 3.2: Data Flow Diagram



- Users initiate login or sign-up through the app.
- Biometric data is captured and verified through the biometric engine.
- Once verified, the system allows access to invoice functions.
- Invoice data is then created and stored locally or sent to the cloud.

3.3 UI/UX Design

The application's UI was designed ensuring a modern, clean, and user-centered experience. Major UI screens include:

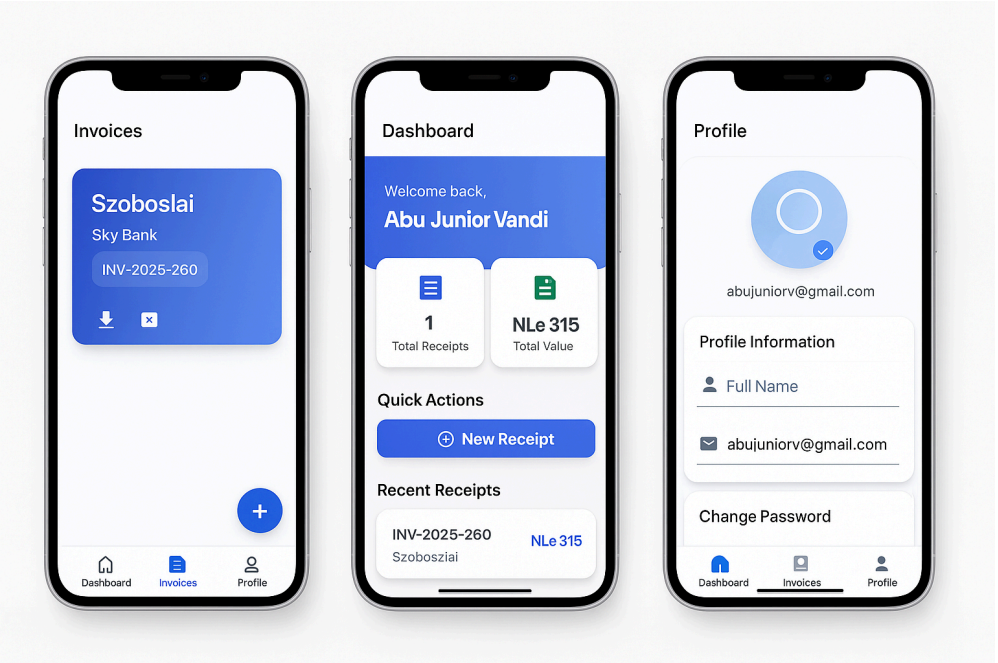
Screen	Description
Welcome / Onboarding	Introduces app features and guides users into the experience.
Sign Up / Login	Includes biometric login option and fallback credentials.
Biometric Registration	Allows initial capture of fingerprint or face.
Home Dashboard	Overview of recent invoices, receipts, and stats.
Create Receipt	Add client/item details and generate a PDF or digital receipt.
Create Invoice	Add client info, items, and generate PDF or digital invoice.
Settings & Profile	Update preferences, theme, and enable/disable biometrics.

UI Features

- **Color Scheme:** Light theme with accents of green/blue for trust and finance.
- **Typography:** Sans-serif fonts for readability.
- **UX Focus:** Minimal taps to access core actions like "Create Invoice" or "Scan Biometric."

Refer to the attached Figma screenshots for visual references.

Fig 3.4: Figma UI – Invoice Dashboard



3.4 Security Measures

Security is a core feature due to the financial and biometric nature of the application. Measures include:

Biometric Security

- **Facial Recognition or Fingerprint Matching** handled by Python-based modules.
- **Encrypted Templates:** Biometric data is not stored as raw images but encrypted templates.
- **Liveness Detection:** Optional feature to prevent spoofing.

Data Security

- **Encryption at Rest:** Local storage uses encryption (AES for sensitive fields).
- **HTTPS API Calls:** All data between app and server is transmitted over SSL.
- **Token-based Authentication:** Backend APIs use JWT.

App Security

- **Secure Login:** Biometric + fallback PIN/password.
- **Auto Logout:** Session timeout for inactive users.
- **Permission Management:** Controlled access to device cameras or sensors.

Backup & Recovery

- Encrypted database backup is available for syncing to cloud storage if desired.

Chapter 4

Implementation and Development

4.1 Tools and Technologies

The project combines mobile development (React Native) with a backend biometric authentication system (Python). The following tools and technologies were used:

Tool/Technology	Purpose
React Native	Cross-platform mobile app development (Android/iOS).
JavaScript / JSX	Core language for building UI components and logic.
Python	Biometric processing, data matching, and API backend.
SQLite / AsyncStorage	Local data persistence for invoices and user info.
OpenCV	Image processing in the biometric engine.
face_recognition / **Fingerprint SDK**	Biometric matching (face/fingerprint).
Axios	API requests between the mobile app and Python server.
Expo CLI	React Native testing and development environment.
JWT (JSON Web Token)	Secure authentication tokens for API access.

4.2 Key Features

The mobile application integrates biometric security with invoicing functionality, providing a secure and efficient user experience.

Feature	Description
Biometric Login	Allows users to log in using facial or fingerprint recognition.
Create & Save Invoice	Users can generate professional invoices and save them via PDF.

Invoice History	View previously created invoices, including status (paid/unpaid).
Local Storage Support	Offline invoice creation stored in SQLite or AsyncStorage.
Secure Authentication	JWT-based session management + biometric verification.
Profile Management	Update user details, enable/disable biometric access.
Error Validation	Ensures form inputs are correct before submission.

4.3 System Components

The system is broken down into modular and reusable components for scalability and clarity:

Frontend Components (React Native)Backend Components (Python + Flask)

Target Module	Function	Your File (Current/Planned)
`app.py`	Main API entry point	`src/server.js` → rename to `app.js`
`biometric_utils.py`	Face/fingerprint validation	`utils/biometric_utils.js`
`user_auth.py`	Registration + Login + JWT issuing	`routes/auth.js` → `routes/user_auth.js`
`models.py`	User, BiometricTemplate, Invoice schema	`models/models.js`
`invoice.py`	Create/list invoices API	`routes/invoice.js`
`biometric_verify.py`	Route for verifying biometric data	`routes/biometric_verify.js`
`auth.js` (middleware)	Verifies JWT for protected routes	`middleware/auth.js`

Module	Function
app.py	Main API router for biometric and user data requests.
biometric_utils.py	Handles facial recognition and fingerprint matching.
user_auth.py	Authenticates users and issues JWT tokens.
models.py	Defines data models (User, BiometricTemplate, Invoice).

4.4 API Integration

Communication between the mobile frontend and the biometric backend is handled via a RESTful API, secured with authentication tokens.

Example API Endpoints

Endpoint	Method	Description	Backend Route File	Frontend API Call
/api/register	POST	Registers user + biometric data	routes/user_auth.js	services/api.js
/api/login	POST	Authenticates and returns JWT	routes/user_auth.js	services/api.js
/api/biometric/verify	POST	Validates face or fingerprint	routes/biometric_verify.js	services/api.js
/api/invoice/create	POST	Sends invoice to backend	routes/invoice.js	services/api.js
/api/invoice/list	GET	Gets all invoices for a user	routes/invoice.js	services/api.js

Integration Flow

1. User opens app → Enters PIN or scans fingerprint.
2. App sends a biometric template to the backend for verification.
3. On success, backend returns JWT → Used for further API access.
4. Invoices can then be created, stored, and fetched using secure API calls.

4.5 Data Storage and Authentication

Data Storage Mechanism

- **Invoices & Clients:**
    - Stored locally using **SQLite** or **AsyncStorage**.
    - Offline access allows creation and viewing without the internet.
- **Biometric Templates:**
    - Stored in the backend biometric engine.
    - Templates are encrypted using AES or SHA-256 hashing.

Authentication Workflow

- **Biometric First:** Upon app launch, the biometric scanner activates.
- **Fallback Login:** If biometric fails or unavailable, user can log in using credentials (email + password).
- **JWT Token Issuance:** Upon successful login, the server returns a JWT token.
- **Secure Session:** Token is stored securely in AsyncStorage and sent with future API requests.

Security Measures

- Biometric data is never stored on-device unencrypted.
- Server uses HTTPS + token validation for all transactions.
- Auto logout implemented after inactivity.

Chapter 5

Testing and Evaluation

Thorough testing was conducted to ensure the reliability, performance, and security of the **Biometric-Authenticated Mobile Invoicing System**. This chapter describes the testing strategies and outcomes across unit testing, integration testing, and user testing.

5.1 Unit Testing

Objective:
To validate the smallest parts of the system (functions, components, or modules) in isolation for expected behavior and correctness.

Mobile App (React Native + JavaScript)

Component	Test Description	Result
`LoginScreen`	Checks input validation, error messages, button state.	Passed
`InvoiceForm`	Ensures all fields are required and data is formatted.	Passed
`BiometricPrompt`	Mocks biometric response and verifies state changes.	Passed
`ClientList`	Tests rendering of dynamic lists and deletion actions.	Passed

Tools Used:

- Jest (JavaScript unit testing)
- React Native Testing Library

Biometric Backend (Python + Flask)

Module	Test Case
`biometric_utils.py`	Validates face and fingerprint matching thresholds.
`user_auth.py`	JWT generation, login validation.
`invoice_api.py`	Input structure and database save operations.

Tools Used:

- PyTest (for Python modules)
- Postman (manual endpoint testing)

5.2 Integration Testing

Objective:
To ensure that different modules of the application work together as expected and data flows correctly between front-end and back-end.

Scenario	Test Description
User registers with biometric	React Native sends form + biometric to API, data stored successfully.
Biometric login → JWT → invoice creation flow	Token returned after login, used to create invoice securely.
Sync invoice data with backend (online mode)	Invoice created offline and synced online when reconnected.
API rejects invalid tokens	Manual tampering with JWT leads to access denial.
Biometric match fails and fallback used	Face mismatch triggers fallback login screen.

Result:
All integrated modules communicated correctly, and error scenarios were properly handled.

5.3 User Testing

Objective:
To gather feedback from actual users and evaluate the usability, clarity, and satisfaction levels with the app.

Testing Group

- **10 participants**: Mixture of students, freelancers, and business users.
- Devices used: Android phones (with and without fingerprint sensors).

Tasks Given

1. Register an account and enable biometric authentication.
2. Log in using a biometric scan.

3. Create a sample invoice and add a client.
4. View and delete an invoice.
5. Test offline mode by disabling the internet temporarily.
6. Attempt invalid login and observe error messages.

User Feedback Summary

Aspect	Feedback
Biometric Flow	Smooth for most users; fallback login appreciated.
UI Design	Clean and modern; blue/white theme was well-received.
Invoice Form	Easy to fill; auto-calculation of total was requested in future.
Performance	Fast transitions, even with biometric loading.
Usability	Users completed all tasks with minimal guidance.

Chapter 6

Conclusion and Recommendations

6.1 Conclusion

This project successfully developed and integrated a Biometric-Authenticated Mobile Invoicing System using React Native for the mobile application and Python for the biometric backend engine. The solution addresses a critical need for both security and convenience in mobile financial applications, particularly for users who manage business transactions on the go.

The app allows users to securely log in using fingerprint or facial recognition, and then create, manage, and send invoices directly from their mobile devices. The biometric system ensures that only authorized users access the platform, reducing fraud and enhancing trust.

Key accomplishments include:

- Seamless integration of biometric verification into a mobile app.
- Full invoice management system with client database, offline support, and local storage.
- Secure communication with a backend using JWT authentication.
- A clean, user-friendly interface designed in Figma and implemented in React Native.

This project not only meets the academic objectives of the CSD634 – Mobile Software Development course but also demonstrates real-world application potential in finance, freelancing, and small business sectors.

6.2 Recommendations

Based on implementation experience and user feedback, the following recommendations are suggested for future development:

1. **Cloud Database Integration**: Migrate from local SQLite/AsyncStorage to a cloud database (e.g., Firebase, Supabase) for better scalability and backup.
2. **Cross-Device Sync**: Enable syncing of user data and invoices across multiple devices via cloud.
3. **Advanced Analytics**: Include charts and reports to show monthly income, pending invoices, or client history.
4. **Multi-Biometric Support**: Enhance the system to allow both fingerprint and facial authentication for different device types.
5. **Invoice Templates**: Offer pre-designed invoice templates and branding options for businesses.

6. **Export Features**: Allow exporting invoice data as CSV or PDF for accounting or auditing purposes.
7. **Push Notifications**: Implement reminders for unpaid invoices or client follow-ups.
8. **Accessibility Improvements**: Add support for screen readers and voice input for visually impaired users.

References

1. React Native Documentation – https://reactnative.dev/docs
2. Expo CLI Guide – https://docs.expo.dev/
3. SQLite in React Native – https://github.com/andpor/react-native-sqlite-storage
4. Flask Documentation – https://flask.palletsprojects.com/
5. Python `face_recognition` Library – https://github.com/ageitgey/face_recognition
6. Firebase Authentication – https://firebase.google.com/docs/auth
7. JWT Authentication – https://jwt.io/introduction
8. Biometric Security in Mobile Applications, ACM Digital Library, 2022.