Healthcare Reception Verification System

Project Overview and Development

Introduction

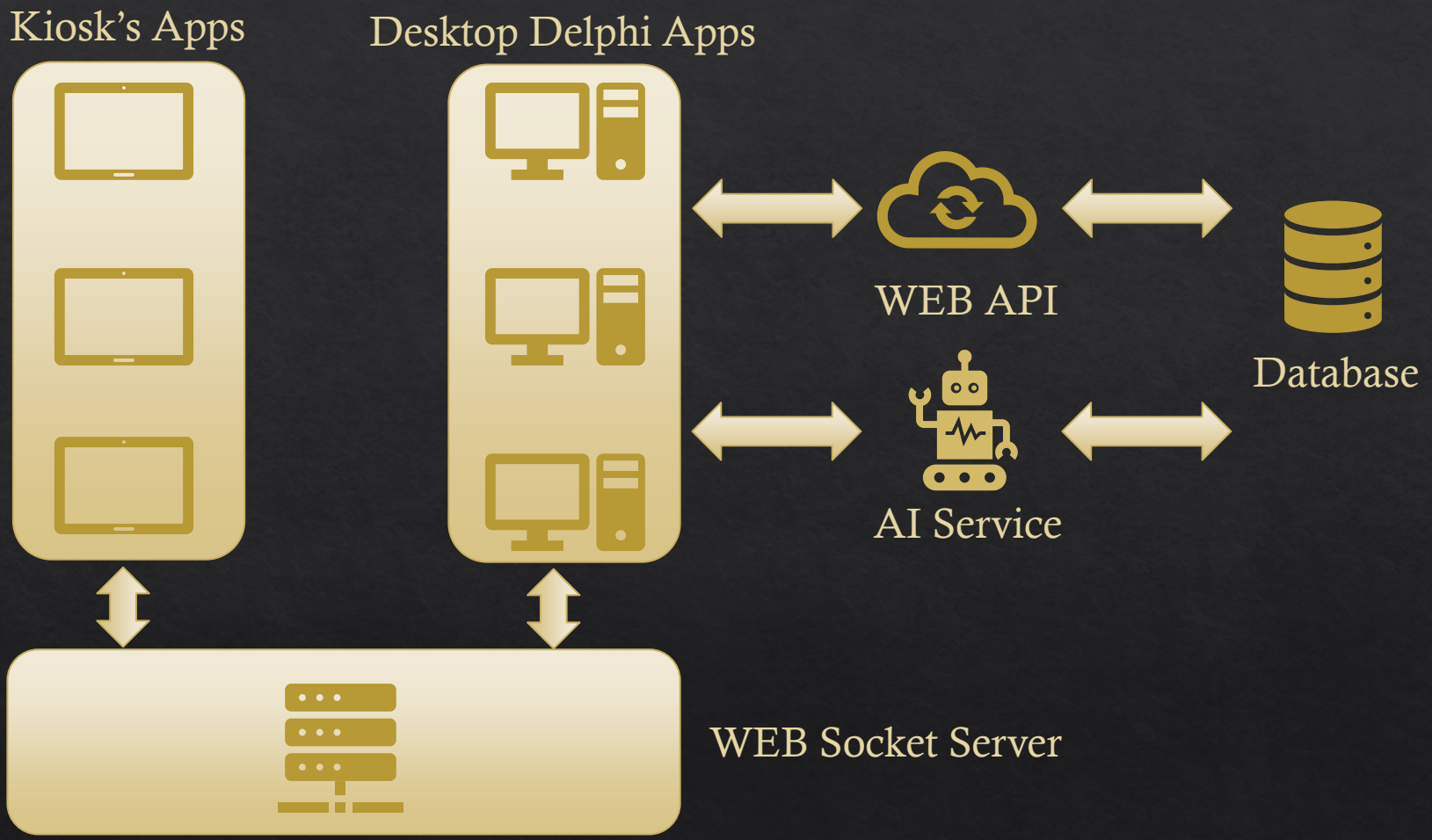
The **Patient Identity Verification & Check-In System** is designed to **streamline and secure the check-in process** for patients at the front desk of a healthcare facility. The system ensures **real-time verification of patient identity** using a **touchscreen kiosk**, allowing multiple patients to verify their identity simultaneously.

This system integrates various technologies, including **Delphi (VCL), React.js, C#/ASP.NET Core, and WebSockets**, to provide **a seamless, efficient, and scalable** solution.

System Architecture Overview

The system follows a **modular and service-oriented architecture**, consisting of the following **four main components**:

- 1. RESTful Web API: Provides access to PostgresSQL database for storing and retrieving data.
- 2. WebSocket Server: Facilitates real-time communication between the desktops and kiosks.
- 3. Verification Kiosk App: A touchscreen web app for patients to complete their identity verification.
- 4. Desktop Delphi App: Used by reception staff to manage check-ins and receive results.


Kiosk's Apps
Desktop Delphi Apps
WEB API
Database
AI Service
WEB Socket Server

1. RESTful Web API (Backend & Data Layer)

- Role: Centralized data access layer that handles database operations securely.
- Responsibilities:
  - ✅ Provide secure access to the PostgresSQL database (via REST API).
  - ✅ Retrieve, modify, and update appointment and verification records.
  - ✅ Enforce business rules and validation on data operations.
  - ✅ Serve as the single source of truth for all data-related queries.
- Technologies: C#/ASP.NET Core, Entity Framework, PostgresSQL Database.
- Communicates with: Delphi Desktop App (via REST API).

2. WebSocket Server (Real-Time Communication)

- Role: Manages instant, bidirectional messaging between apps, ensuring real-time updates.
- Responsibilities:
  - ✅ Notify Delphi Apps when verification results are available.
  - ✅ Notify other Delphi Apps if an appointment status changes.
  - ✅ Send verification requests to Kiosk Apps when triggered by reception staff.
  - ✅ Receive verification results from Kiosk Apps and broadcast them to reception staff.
- Technologies: Node.js, WebSocket Library.
- Communicates with: Delphi Desktop App (for real-time updates), Kiosk App (for verification requests and results).

3. Kiosk App (Patient Verification Terminal)

- Role: Provides a touchscreen interface for patients to verify their identity.
- Responsibilities:
  - ✅ Wait for a verification request from WebSocket Server.
  - ✅ Display security questions to the patient, with four possible responses.
  - ✅ Collect responses and send verification results to WebSocket Server.
  - ✅ Update kiosk status (Available, In Use, Offline).
- Technologies: React.js (Frontend), WebSockets (Real-Time Messaging).
- Communicates with: WebSocket Server.

4. Delphi Desktop App (Reception Front Desk)

- Role: The central control panel for reception staff to manage patient check-in and verification.
- Responsibilities:
  - ☑ Display appointments, grouped by status (Pending, Canceled, Completed).
  - ☑ Modify appointment statuses as needed.
  - ☑ Retrieve and display the list of available kiosks and their statuses.
  - ☑ Initiate a patient verification request on an available kiosk.
  - ☑ Receive real-time updates on verification results and update the database.
  - ☑ Stay synchronized with other reception Delphi apps when data changes.
- Technologies: Embarcadero Delphi 11 (VCL, FireDAC for DB access, REST API Client, WebSocket Client).
- Communicates with: Web API (for data), WebSocket Server (for real-time updates and verification requests).

Key Features and Benefits

✅ **Modular Design:** Each component is independent, making the system scalable and maintainable.

✅ **Separation of Concerns (SoC):**

- Delphi App (UI & Business Logic)
- Web API (Data Access & Validation)
- WebSocket Server (Real-Time Communication)
- Kiosk App (User Interaction)
  Secure Database Access: Only the **Web API** accesses the MS-SQL database.
  Event-Driven Real-Time Communication: WebSockets ensure **instant UI updates** without polling.
  Scalability & Extensibility: Easily add more kiosks or reception workstations.
- Web API can support future mobile/web clients.
  Cross-Technology Integration: Delphi, .NET, React.js, and Node.js work together efficiently.

Summary

- The system follows enterprise-level best practices by ensuring modular design, real-time communication, secure database access, and scalable architecture.
- It solves a real-world problem in healthcare by automating patient check-in and verification, improving efficiency.
- The Delphi Desktop App is a core component, integrating seamlessly with modern web technologies like React.js, C# Web API, and WebSockets.
- The architecture ensures high availability, reliability, and scalability, making it future-proof for additional integrations.