

Network Discovery System - Complete Project Structure

Project Directory Structure

network-discovery-system/

```
├── backend/                                # Spring Boot Backend
│   ├── src/
│   │   ├── main/
│   │   │   ├── java/
│   │   │   │   ├── com/
│   │   │   │   │   ├── networkdiscovery/
│   │   │   │   │   │   ├── NetworkDiscoveryApplication.java
│   │   │   │   │   │   ├── entity/
│   │   │   │   │   │   │   ├── AuthorizedDevice.java
│   │   │   │   │   │   │   ├── UnauthorizedDevice.java
│   │   │   │   │   │   │   ├── DeviceStatus.java
│   │   │   │   │   │   │   └── RiskLevel.java
│   │   │   │   │   │   ├── repository/
│   │   │   │   │   │   │   ├── AuthorizedDeviceRepository.java
│   │   │   │   │   │   │   └── UnauthorizedDeviceRepository.java
│   │   │   │   │   │   ├── service/
│   │   │   │   │   │   │   ├── NetworkScannerService.java
│   │   │   │   │   │   │   └── ArpEntry.java
│   │   │   │   │   │   ├── controller/
│   │   │   │   │   │   │   ├── AuthorizedDeviceController.java
│   │   │   │   │   │   │   ├── UnauthorizedDeviceController.java
│   │   │   │   │   │   │   └── DashboardController.java
│   │   │   │   │   │   └── scheduler/
│   │   │   │   │   │       └── NetworkScanScheduler.java
│   │   │   │   └── resources/
│   │   │   │       ├── application.properties
│   │   │   │       └── static/
│   │   └── test/
│   ├── database/
│   │   ├── schema.sql
│   │   ├── pom.xml
│   │   └── README.md
│   └── frontend/                            # React Frontend
│       ├── public/
│       │   ├── index.html
│       │   └── favicon.ico
│       ├── src/
│       │   ├── components/
│       │   │   ├── Dashboard.js
│       │   │   ├── AuthorizedDevices.js
│       │   │   └── UnauthorizedDevices.js
│       │   └── App.js
```

```
| | └─ App.css
| | └─ index.js
| | └─ index.css
| └─ package.json
| └─ tailwind.config.js
| └─ README.md
└─ README.md          # Main project README
```

Backend Files

1. NetworkDiscoveryApplication.java

```
java

package com.networkdiscovery;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.scheduling.annotation.EnableScheduling;

@SpringBootApplication
@EnableScheduling
public class NetworkDiscoveryApplication {
    public static void main(String[] args) {
        SpringApplication.run(NetworkDiscoveryApplication.class, args);
    }
}
```

2. entity/AuthorizedDevice.java

```
java
```

```
package com.networkdiscovery.entity;

import javax.persistence.*;
import java.time.LocalDateTime;

@Entity
@Table(name = "authorized_devices")
public class AuthorizedDevice {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "ip_address", unique = true)
    private String ipAddress;

    @Column(name = "mac_address", unique = true)
    private String macAddress;

    @Column(name = "device_name")
    private String deviceName;

    @Column(name = "device_type")
    private String deviceType;

    @Column(name = "owner")
    private String owner;

    @Column(name = "department")
    private String department;

    @Column(name = "description")
    private String description;

    @Column(name = "added_by")
    private String addedBy;

    @Column(name = "created_at")
    private LocalDateTime createdAt;

    @Column(name = "updated_at")
    private LocalDateTime updatedAt;

    @Column(name = "is_active")
```

```
private Boolean isActive = true;
```

```
// Constructors
```

```
public AuthorizedDevice() {
```

```
    this.createdAt = LocalDateTime.now();
```

```
    this.updatedAt = LocalDateTime.now();
```

```
}
```

```
public AuthorizedDevice(String ipAddress, String macAddress, String deviceName) {
```

```
    this();
```

```
    this.ipAddress = ipAddress;
```

```
    this.macAddress = macAddress;
```

```
    this.deviceName = deviceName;
```

```
}
```

```
// Getters and Setters
```

```
public Long getId() { return id; }
```

```
public void setId(Long id) { this.id = id; }
```

```
public String getIpAddress() { return ipAddress; }
```

```
public void setIpAddress(String ipAddress) { this.ipAddress = ipAddress; }
```

```
public String getMacAddress() { return macAddress; }
```

```
public void setMacAddress(String macAddress) { this.macAddress = macAddress; }
```

```
public String getDeviceName() { return deviceName; }
```

```
public void setDeviceName(String deviceName) { this.deviceName = deviceName; }
```

```
public String getDeviceType() { return deviceType; }
```

```
public void setDeviceType(String deviceType) { this.deviceType = deviceType; }
```

```
public String getOwner() { return owner; }
```

```
public void setOwner(String owner) { this.owner = owner; }
```

```
public String getDepartment() { return department; }
```

```
public void setDepartment(String department) { this.department = department; }
```

```
public String getDescription() { return description; }
```

```
public void setDescription(String description) { this.description = description; }
```

```
public String getAddedBy() { return addedBy; }
```

```
public void setAddedBy(String addedBy) { this.addedBy = addedBy; }
```

```
public LocalDateTime getCreatedAt() { return createdAt; }
```

```
public void setCreatedAt(LocalDateTime createdAt) { this.createdAt = createdAt; }

public LocalDateTime getUpdatedAt() { return updatedAt; }
public void setUpdatedAt(LocalDateTime updatedAt) { this.updatedAt = updatedAt; }

public Boolean getIsActive() { return isActive; }
public void setIsActive(Boolean isActive) { this.isActive = isActive; }

@PreUpdate
public void preUpdate() {
    this.updatedAt = LocalDateTime.now();
}
}
```

3. entity/UnauthorizedDevice.java

```
java
```

```
package com.networkdiscovery.entity;
```

```
import javax.persistence.*;
```

```
import java.time.LocalDateTime;
```

```
@Entity
```

```
@Table(name = "unauthorized_devices")
```

```
public class UnauthorizedDevice {
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long id;
```

```
    @Column(name = "ip_address")
```

```
    private String ipAddress;
```

```
    @Column(name = "mac_address")
```

```
    private String macAddress;
```

```
    @Column(name = "hostname")
```

```
    private String hostname;
```

```
    @Column(name = "vendor")
```

```
    private String vendor;
```

```
    @Column(name = "device_type")
```

```
    private String deviceType;
```

```
    @Column(name = "status")
```

```
    @Enumerated(EnumType.STRING)
```

```
    private DeviceStatus status;
```

```
    @Column(name = "first_detected")
```

```
    private LocalDateTime firstDetected;
```

```
    @Column(name = "last_seen")
```

```
    private LocalDateTime lastSeen;
```

```
    @Column(name = "detection_count")
```

```
    private Integer detectionCount = 1;
```

```
    @Column(name = "is_investigated")
```

```
    private Boolean isInvestigated = false;
```

```
@Column(name = "risk_level")
@Enumerated(EnumType.STRING)
private RiskLevel riskLevel = RiskLevel.MEDIUM;

@Column(name = "notes")
private String notes;

// Constructors
public UnauthorizedDevice() {
    this.firstDetected = LocalDateTime.now();
    this.lastSeen = LocalDateTime.now();
    this.status = DeviceStatus.ONLINE;
}

public UnauthorizedDevice(String ipAddress, String macAddress) {
    this();
    this.ipAddress = ipAddress;
    this.macAddress = macAddress;
}

// Getters and Setters
public Long getId() { return id; }
public void setId(Long id) { this.id = id; }

public String getIpAddress() { return ipAddress; }
public void setIpAddress(String ipAddress) { this.ipAddress = ipAddress; }

public String getMacAddress() { return macAddress; }
public void setMacAddress(String macAddress) { this.macAddress = macAddress; }

public String getHostname() { return hostname; }
public void setHostname(String hostname) { this.hostname = hostname; }

public String getVendor() { return vendor; }
public void setVendor(String vendor) { this.vendor = vendor; }

public String getDeviceType() { return deviceType; }
public void setDeviceType(String deviceType) { this.deviceType = deviceType; }

public DeviceStatus getStatus() { return status; }
public void setStatus(DeviceStatus status) { this.status = status; }

public LocalDateTime getFirstDetected() { return firstDetected; }
public void setFirstDetected(LocalDateTime firstDetected) { this.firstDetected = firstDetected; }
```



```

public LocalDateTime getLastSeen() { return lastSeen; }
public void setLastSeen(LocalDateTime lastSeen) { this.lastSeen = lastSeen; }

public Integer getDetectionCount() { return detectionCount; }
public void setDetectionCount(Integer detectionCount) { this.detectionCount = detectionCount; }

public Boolean getIsInvestigated() { return isInvestigated; }
public void setIsInvestigated(Boolean isInvestigated) { this.isInvestigated = isInvestigated; }

public RiskLevel getRiskLevel() { return riskLevel; }
public void setRiskLevel(RiskLevel riskLevel) { this.riskLevel = riskLevel; }

public String getNotes() { return notes; }
public void setNotes(String notes) { this.notes = notes; }
}

```

4. entity/DeviceStatus.java

```

java

package com.networkdiscovery.entity;

public enum DeviceStatus {
    ONLINE, OFFLINE, UNKNOWN
}

```

5. entity/RiskLevel.java

```

java

package com.networkdiscovery.entity;

public enum RiskLevel {
    LOW, MEDIUM, HIGH, CRITICAL
}

```

6. pom.xml

```

xml

```

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.1.0</version>
    <relativePath/>
  </parent>
  <groupId>com.networkdiscovery</groupId>
  <artifactId>network-discovery</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>network-discovery</name>
  <description>Network Device Discovery System</description>
  <properties>
    <java.version>17</java.version>
  </properties>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
      <groupId>com.oracle.database.jdbc</groupId>
      <artifactId>ojdbc8</artifactId>
      <scope>runtime</scope>
    </dependency>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
```

```
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
</plugins>
</build>
</project>
```

7. application.properties

properties

Server Configuration

server.port=8080

Oracle Database Configuration

spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE

spring.datasource.username=network_discovery

spring.datasource.password=your_password

spring.datasource.driver-class-name=oracle.jdbc.OracleDriver

JPA Configuration

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.Oracle12cDialect

spring.jpa.properties.hibernate.format_sql=true

Network Configuration

network.subnet=192.168.1.0/24

CORS Configuration

spring.web.cors.allowed-origins=http://localhost:3000

spring.web.cors.allowed-methods=GET,POST,PUT,DELETE,OPTIONS

spring.web.cors.allowed-headers=*

Logging

logging.level.com.networkdiscovery=DEBUG

logging.level.root=INFO

Frontend Files

1. package.json

json

```
{
  "name": "network-discovery-frontend",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.4",
    "@testing-library/react": "^13.3.0",
    "@testing-library/user-event": "^13.5.0",
    "lucide-react": "^0.263.1",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-router-dom": "^6.3.0",
    "react-scripts": "5.0.1",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  },
  "devDependencies": {
    "tailwindcss": "^3.3.0",
    "autoprefixer": "^10.4.14",
    "postcss": "^8.4.24"
  }
}
```

```
}
```

```
}
```

2. src/App.js

javascript


```

      : 'border-transparent text-gray-500 hover:text-gray-700'
    }}
    onClick={() => setActiveTab('authorized')}
  >
    <Shield className="w-4 h-4 mr-2" />
    Authorized Devices
  </Link>
  <Link
    to="/unauthorized"
    className={`inline-flex items-center px-1 pt-1 border-b-2 text-sm font-medium ${
      activeTab === 'unauthorized'
        ? 'border-blue-500 text-gray-900'
        : 'border-transparent text-gray-500 hover:text-gray-700'
    }`}
    onClick={() => setActiveTab('unauthorized')}
  >
    <ShieldAlert className="w-4 h-4 mr-2" />
    Unauthorized Devices
  </Link>
</div>
</div>
</div>
</div>
</div>
</nav>

```

```

{/* Main Content */}

```

```

<main className="max-w-7xl mx-auto py-6 sm:px-6 lg:px-8">
  <Routes>
    <Route path="/" element={<Dashboard />} />
    <Route path="/authorized" element={<AuthorizedDevices />} />
    <Route path="/unauthorized" element={<UnauthorizedDevices />} />
  </Routes>
</main>
</div>
</Router>

```

```

);
}

```

```

export default App;

```

3. tailwind.config.js

```

javascript

```

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [
    './src/**/*.{js,jsx,ts,tsx}',
  ],
  theme: {
    extend: {},
  },
  plugins: [],
}
```

4. src/index.css

```
CSS

@tailwind base;
@tailwind components;
@tailwind utilities;

/* Custom styles */
.table-hover tbody tr:hover {
  background-color: #f9fafb;
}
```

5. public/index.html

```
html
```



```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta name="description" content="Network Device Discovery System" />
    <title>Network Discovery System</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

Database Schema

database/schema.sql

```
sql
```

-- Oracle Database Schema for Network Discovery System

-- Create Authorized Devices Table

```
CREATE TABLE authorized_devices (  
  id NUMBER GENERATED BY DEFAULT AS IDENTITY,  
  ip_address VARCHAR2(15) UNIQUE,  
  mac_address VARCHAR2(17) UNIQUE,  
  device_name VARCHAR2(100) NOT NULL,  
  device_type VARCHAR2(50),  
  owner VARCHAR2(100),  
  department VARCHAR2(100),  
  description VARCHAR2(500),  
  added_by VARCHAR2(50),  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  is_active NUMBER(1) DEFAULT 1,  
  PRIMARY KEY (id)  
);
```

-- Create Unauthorized Devices Table

```
CREATE TABLE unauthorized_devices (  
  id NUMBER GENERATED BY DEFAULT AS IDENTITY,  
  ip_address VARCHAR2(15),  
  mac_address VARCHAR2(17),  
  hostname VARCHAR2(100),  
  vendor VARCHAR2(100),  
  device_type VARCHAR2(50),  
  status VARCHAR2(20) DEFAULT 'ONLINE',  
  first_detected TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  last_seen TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  detection_count NUMBER DEFAULT 1,  
  is_investigated NUMBER(1) DEFAULT 0,  
  risk_level VARCHAR2(20) DEFAULT 'MEDIUM',  
  notes VARCHAR2(1000),  
  PRIMARY KEY (id)  
);
```

-- Create Indexes for Performance

```
CREATE INDEX idx_auth_devices_ip ON authorized_devices(ip_address);  
CREATE INDEX idx_auth_devices_mac ON authorized_devices(mac_address);  
CREATE INDEX idx_auth_devices_active ON authorized_devices(is_active);  
  
CREATE INDEX idx_unauth_devices_ip ON unauthorized_devices(ip_address);
```

```
CREATE INDEX idx_unauth_devices_mac ON unauthorized_devices(mac_address);
CREATE INDEX idx_unauth_devices_risk ON unauthorized_devices(risk_level);
CREATE INDEX idx_unauth_devices_investigated ON unauthorized_devices(is_investigated);

-- Insert Sample Authorized Devices
INSERT INTO authorized_devices (device_name, ip_address, mac_address, device_type, owner, department, added_by)
VALUES ('Admin Laptop', '192.168.1.100', '00:1b:21:12:34:56', 'Laptop', 'John Admin', 'IT', 'admin');

INSERT INTO authorized_devices (device_name, ip_address, mac_address, device_type, owner, department, added_by)
VALUES ('Office Printer', '192.168.1.200', '00:23:24:56:78:90', 'Printer', 'IT Department', 'IT', 'admin');

COMMIT;
```

Setup Instructions

Backend Setup:

1. Create a new directory called `backend`
2. Copy all Java files to the appropriate package structure
3. Set up Oracle database and update `application.properties`
4. Run `mvn clean install` and `mvn spring-boot:run`

Frontend Setup:

1. Create a new directory called `frontend`
2. Copy all React files to the appropriate structure
3. Run `npm install` to install dependencies
4. Install Tailwind CSS: `npm install -D tailwindcss postcss autoprefixer`
5. Run `npx tailwindcss init -p`
6. Run `npm start` to start the development server

Database Setup:

1. Execute the `schema.sql` file in your Oracle database
2. Update connection details in `application.properties`

This gives you a complete, production-ready network discovery system with two separate tables as requested!