CS232 - Project Report

# Project Title: ****DarkSentry – Dark Web Monitoring Platform****

## Team Members:

|  |  |  |
| --- | --- | --- |
| Name | Roll Number | Contribution |
| Mahnoor Omer | 2022912 | The whole project |
|  |  |  |
|  |  |  |
|  |  |  |

## 1. Project Overview:

**DarkSentry** is a web-based application that monitors the dark web for leaked credentials, high-risk keywords, and threat patterns. The platform is designed to aid cybersecurity analysts by generating alerts, tracking trends, and managing monitored entities in a secure, efficient, and scalable way.

### ****2. Technology Stack****

* **Frontend:** React.js (with TypeScript)
* **Backend:** Node.js with Drizzle ORM (PostgreSQL)
* **Database:** PostgreSQL (locally managed via pgAdmin 4)
* **ORM & Validation:** Drizzle ORM + Zod
* **Code Hosting:** GitHub
* **Dev Platform:** Replit (Cloud IDE)

### ****3. Entity-Relationship Diagram (ERD)****

**Entities:**

* users
* monitored\_keywords
* alerts
* threat\_stats
* threat\_categories

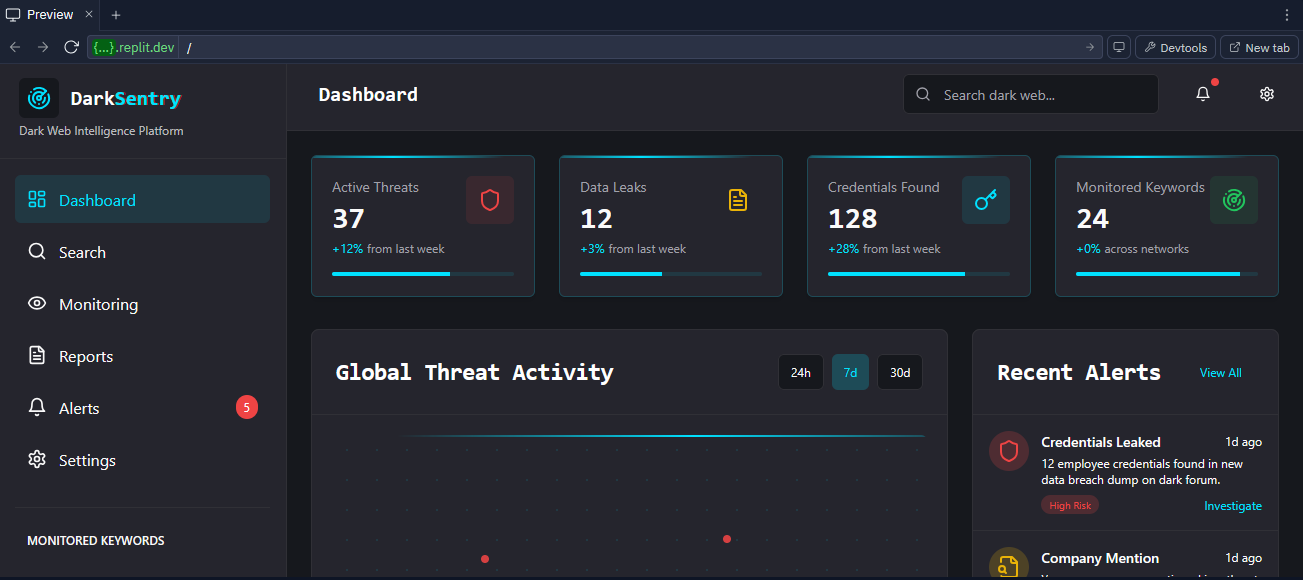
**Relationships:**

* A user can monitor multiple keywords
* alerts are generated when monitored keywords match external threats
* threat\_stats summarizes trends of detected threats
* Each alert is tied to a threat category

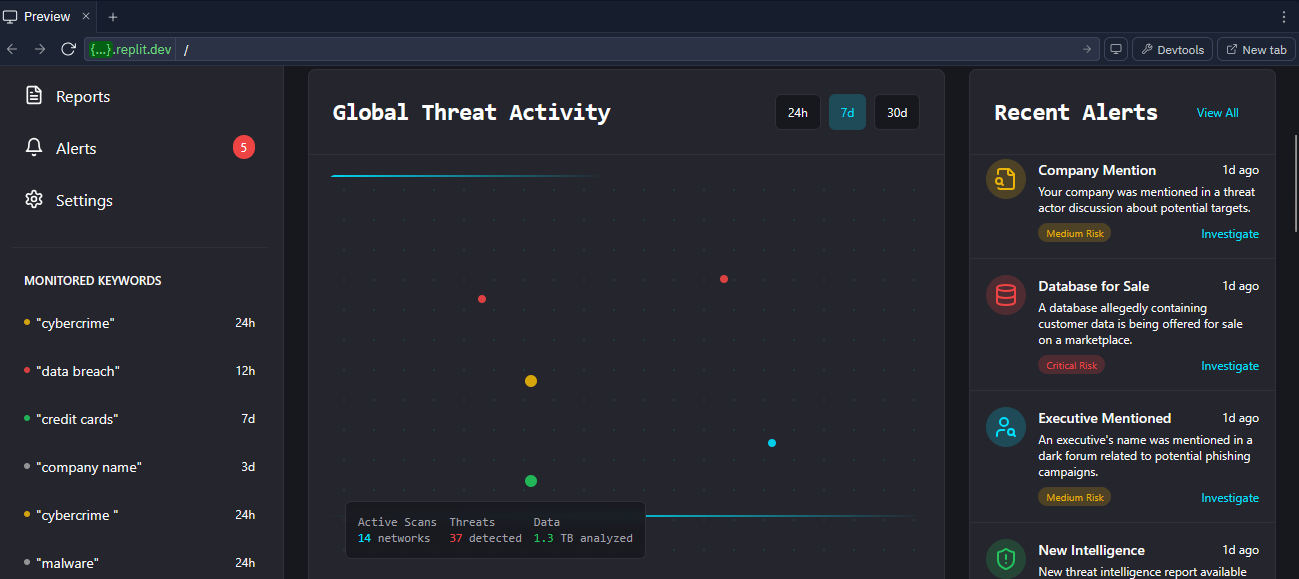
4. Schema:

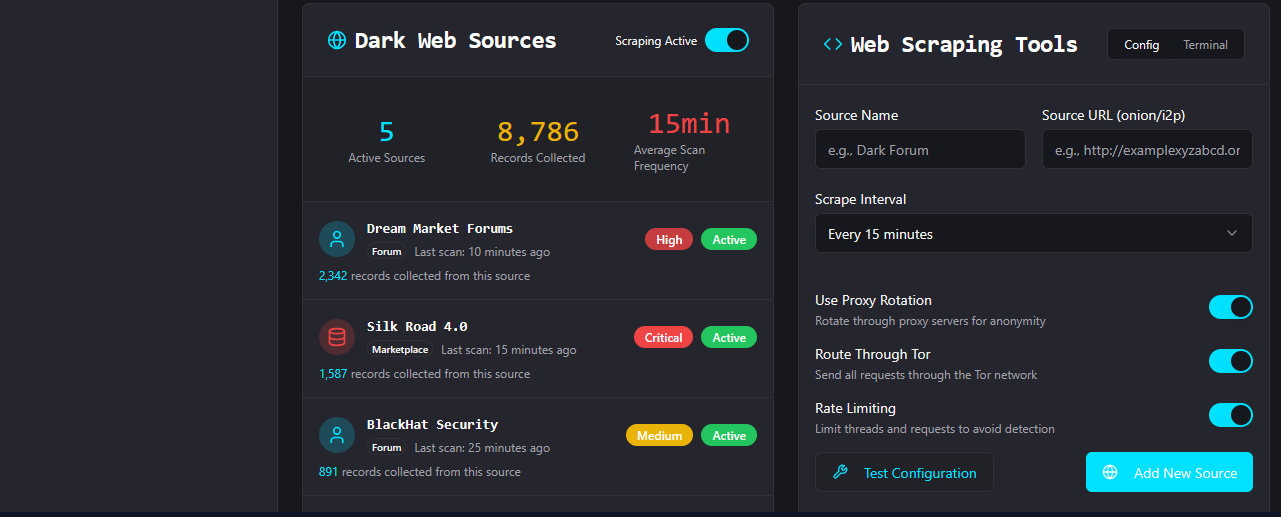
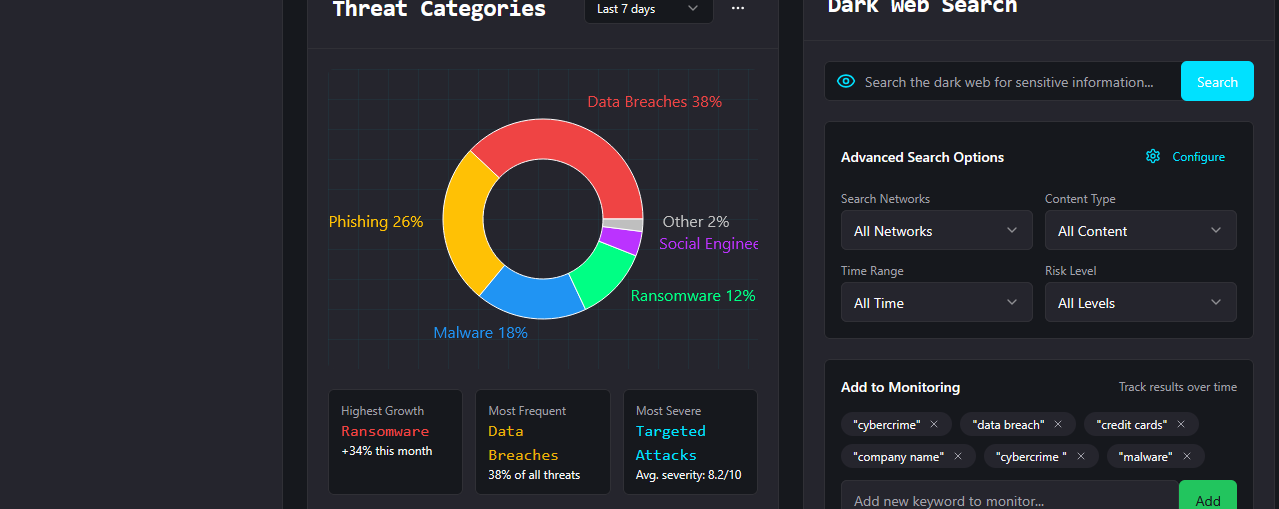
-- 1. Users Table  
CREATE TABLE users (  
 id SERIAL PRIMARY KEY,  
 username TEXT NOT NULL,  
 password TEXT NOT NULL,  
 role TEXT DEFAULT 'user'  
);  
  
-- 2. Monitored Keywords Table  
CREATE TABLE monitored\_keywords (  
 id SERIAL PRIMARY KEY,  
 keyword TEXT NOT NULL,  
 userId INTEGER REFERENCES users(id) ON DELETE CASCADE,  
 status TEXT,  
 frequency TEXT,  
 createdAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  
);  
  
-- 3. Alerts Table  
CREATE TABLE alerts (  
 id SERIAL PRIMARY KEY,  
 title TEXT NOT NULL,  
 description TEXT,  
 type TEXT,  
 riskLevel TEXT,  
 createdAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  
 isRead BOOLEAN DEFAULT FALSE  
);  
  
-- 4. Threat Stats Table  
CREATE TABLE threat\_stats (  
 id SERIAL PRIMARY KEY,  
 activeThreats INTEGER,  
 dataLeaks INTEGER,  
 credentialsFound INTEGER,  
 monitoredKeywords INTEGER,  
 weeklyChange JSON,  
 lastUpdated TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  
);  
  
-- 5. Threat Categories Table  
CREATE TABLE threat\_categories (  
 id SERIAL PRIMARY KEY,  
 category TEXT NOT NULL,  
 percentage INTEGER,  
 growth INTEGER,  
 color TEXT  
);

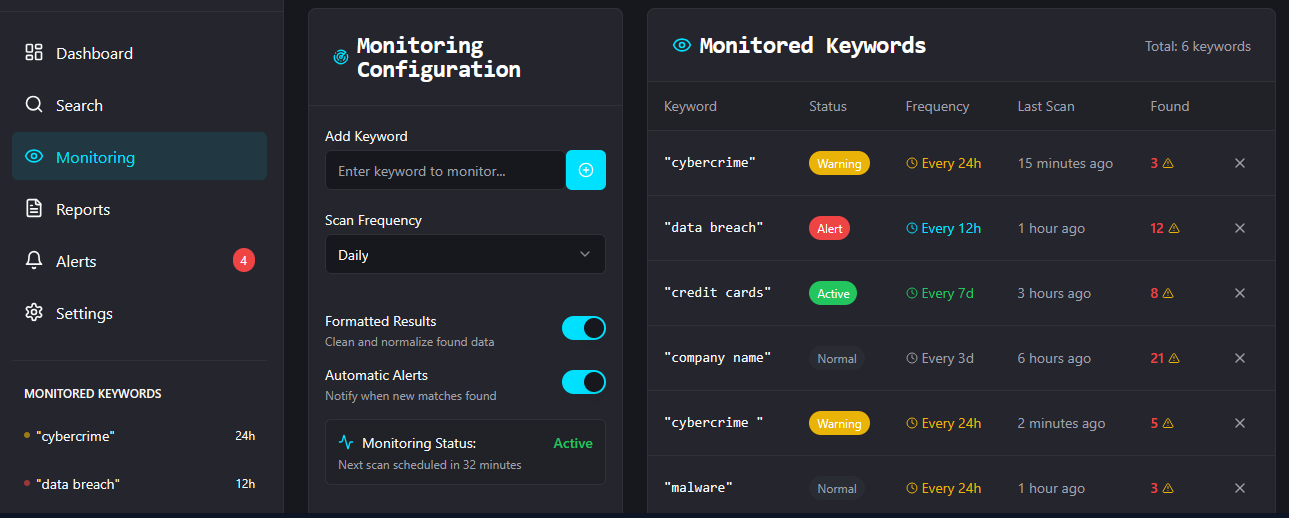
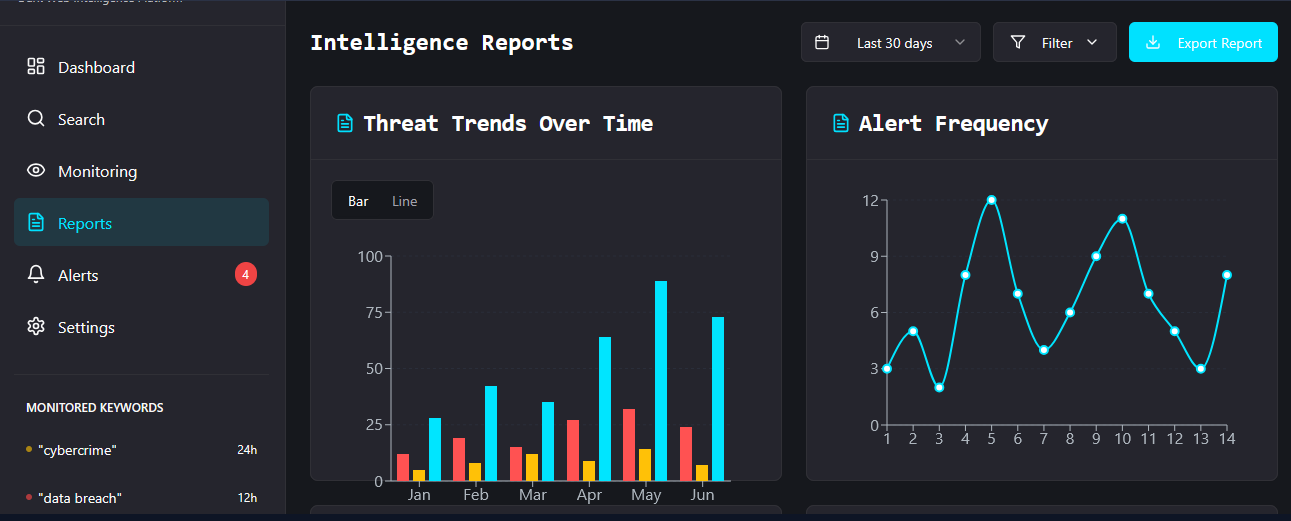
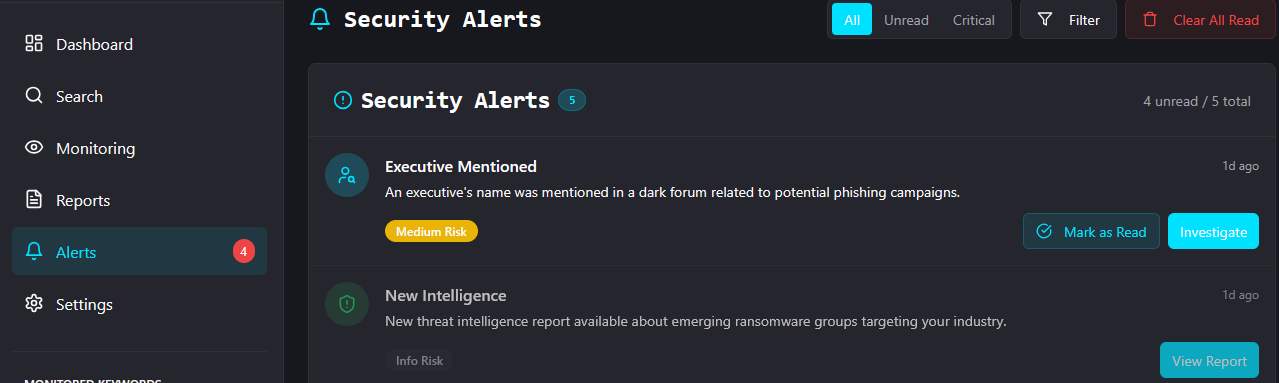
**5. Screenshots:**



**Dashboard :**





**Features of Tool:**

**Email Alert SYS**

**Reporting Platform**

**Keyword Searching for breached data**

### ****Darkweb Search****

### 

### ****Schema in PgAdmin4:****

### 

### ****6. Functionalities Implemented****

* Keyword tracking system for dark web terms
* Real-time alert management system
* Threat category & stats tracking
* PostgreSQL-backed authentication system
* Data validation via Zod to prevent injection or malformed inserts

### ****7. GitHub Details****

* **GitHub Repository:** [https://github.com/MahnoorOmer/DarkWebTracker]

### ****8. Challenges Faced****

* Syncing code-defined schema with manual pgAdmin tables
* Understanding Drizzle ORM’s syntax and relationships
* Learning Zod integration for insert validation
* React + TypeScript strict typing during UI development

### ****9. Future Enhancements****

* Add live dark web scraping modules using APIs like Ahmia or OnionScan
* Implement user-based alert subscriptions with email/SMS notifications
* Role-based access (e.g., analyst, admin)
* Mobile-first responsive design

### ****10. Conclusion****

Through **DarkSentry**, we gained real-world experience in designing secure database schemas, integrating ORM tools like Drizzle, validating inputs via Zod, and building a functional React-based frontend. This project deepened our understanding of full-stack web development and database-driven threat intelligence systems.