THE SAGE GROUP

Mentoring Solutions for Law Enforcement

"Solutions that Empower your Workforce"

Mobile Applications

Overview and Solutions

From First Responder to Crime Reporting, Law Enforcement agencies are relying more on Mobile based applications to enhance and aide in their overall job performance.

The demand for real-time-access, always on availability, fewer delays and faster execution drive organizations to explore custom mobile solutions based around business processes/workflows, visualization tools, and composite applications that blend date from multiple back-end sources.

The Sage Group' project directed mentored approach combines the best of education/training and consulting practices into a hybrid solution that will keep your organization self-sufficient for years. Our experience and capabilities in the mobile app development and wireless technologies have accumulated over the years of working with clients on diverse engagements.

These include the following:

Technology	Capability
Devices	Android, iPhone, iPad, Windows Phone, Blackberry
Operating Systems	Android, iOS, RIM OS, Windows Phone
Tools and SDK's	Sencha
Accelerators	Titanium, PhoneGap, Xamarin
Mobile Enterprise Application Platform	Sybase Unwired Platform
Wireless	GPS, WAP, GRPS, EDGE, HSPA, LTE, Bluetooth, IrDA, Wi-Fi
Databases	SQL Anywhere, SQL Server Compact
Related	QR Codes, NFC

Usage	Third Party Tools Incorporated
GIS Locator	
Speech Recognition & Audio Capture	
Local Storage Utilization	
Crime Report Mapping	
Agency Alert Notifications	
Calendar & Task Collaboration	
Video Capture	

Areas of Expertise & Experience

- Building a Plug & Play Mobile Application Framework
- Use case Development
- Crime Detection Applications
- Mobile Applications Best Practices within Law Enforcement
- Mobile Application Security Best Practices
- Mobile Application Secure Code

	Mobile Security Best Practices
1	Use native SSL libraries on the OS. Some third party libraries have vulnerabilities and can be exploited for man-in-the-middle-attacks
2	Use mutual authentication in apps to validate server connections to ensure that the app is communicating to the server you expect and not a man-in-the-middle.
3	All app communication should be encrypted. Do not disable this in iOS 9.
4	Pin certificates use for encrypted communications and mutual authentication. Do not rely on root certificates stored in the OS, as new roots can be added, which can lead to man-in-the-middle attacks.
5	Avoid using precompiled third party libraries, since you do not know what they do. These could be ad libraries, encryption libraries and graphics libraries. Do not download libraries from non-trusted parties.
6	Only enable inter-app communication for apps you trust must communicate with.
7	Endure inter-app communication is encrypted.
8	Do not store files unencrypted. Ensure that encryption libraries are fully utilized.
9	Do not store non-essential personally-identifiable information inside your app. It is better to download this from servers as needed.
10	Do not store passwords on the device. If you must, store only a hash of the password.
11	iOS apps should store secrets and credentials in the KeyChain. This leverages the security that is built into the KeyChain.
12	Do not send confidential information via SMS of APNS messages. They can be read by anyone who has access to the phone, even if the phone is locked.
13	Be wary of any plug-ins that your app uses. These are often vectors for introducing security vulnerabilities into apps.
14	Only use code and development tools from trusted vendors, and only download core development libraries and tools from Apple or Google's actual download sites. Do not download development tools from unofficial sites, or your app could be infected with malware such XcodeGhost.
15	Enable Position Independent Execution when compiling your app. PIE is important to reduce the chance malicious apps and tools can access known memory locations.
16	Only declare permissions that you actually need and use in the app Do not just copy a permissions list from a generic app.
17	Be very wary of embedding API keys into your app where the API can be used to access sensitive data or accounts on cloud services. Assume API keys have been copied. Ensure that access controls are enforced by the addition of a password or other user credential that is entered by the user and not common to all apps.
18	Consider adding code to check for jailbroken and rooted phones, and not allow transactions from compromised phones/code reviews by team or outside resources.
19	Have a privacy policy that accurately describes what your app and your servers do with data. Have your privacy policy reviewed by internal legal counsel. Ensure that your privacy policy is published with your app and linked to your app store entry.
20	Analyze your app with reputable mobile defense tools to validate is behavior prior to release.