Software Architecture Documentation(uWatch digital forensic tool)

Group Name: MPHETamines

Version: 1.3

Taariq Ghoord	10132806
Martha Mohlala	10353403
Phethile Mkhabela	12097561
Sboniso Masilela	10416260
Harrison Maphuti Setati	12310043

Git repository link:
`https://github.com/MPHETamines/MPHETamines/`

Date: 22/09/2015

Contents

1 Introduction

This section deals with the software architecture requirements of the uWatch system being designed. It handles all aspects of the system's design which form part of its non-functional requirements, in particular the requirements of the software architecture on which the application's functional aspects are developed.This includes:

- The architectural scope.
- Quality requirements.
- The integration and access channel requirements.
- The architectural constraints.
- Architectural patterns and styles used.
- The architectural tactics and strategies used.
- The use of reference architectures and frameworks.
- Access and integration channels.
- Technologies used.

2 Architecture requirements

2.1 Architectural scope

This system integrate the back-end and the front-end. The front-end of the system is a hybrid mobile app which implements all the front end functionalities required. Functionalitites such as capture videos, audios and images and send those files to a server running on the back end. On the back-end the system runs Apache server, where all the large data files are stored, relational database, where user details and the links to where the files users uploaded. This front-end-back-end integration is made possible by PHP scripts on the server. The app makes RESTful request(http) to send files and user details upon registration to the server.

2.2 Quality requirements

- Security(core)
  - The system must prevent sql injection
    - The system does string escaping and input validation
  - The system must verify user emails
    - Upon registration, the user is send OTP to verify hi/her email address
  - The system must implement strong authentication
    - Something user has and something user knows
  - The system must detect attacks from unwanted and unauthorized users.
    - Prevent all unauthorized users from accessing the system
  - The system must resist attacks from unwanted and unauthorized users.
    - Ensure user's password is longer and strong
  - The system must recover from attack from unwanted and unauthorized users.
  - The system must minimize access and permissions given to users who do not have the required privileges.
    - Users can only capture and upload, they can not download what the have uploaded.
  - All communication of sensitive data must be done securely through encryption and secure channels.
    - Protect user's details and files they have uploaded
  - System ensures data integrity
    - Ensure user's details and file are not tempered with
- Scalability(core)
  - The system operate effectively under a load of registered users.
  - The system scales out resources.
  - The system manages resource demand.
- Reliability(core)

  - The system must prevent faults.
  - The system must detect faults.
  - The system must recover from faults.
- Integrability(core)
  - The system must integrate with law enforcement server.
  - The system integrate with Google maps for Geo-location.
- Performance(core)
  - Throughput:The rate at which incoming requests are completed.
  - The system minimize the responsive time.
- Usability(core)
  - The system must be efficient.
  - The system must be easy to use.
  - The user must be satisfied by the system.
- Maintainability
  - System must be easy to be tested, updated and changed if needed.
- Deployability(important)
  - The system is deployed into cross platforms.

2.3 How we intend to achieve Quality requirements

- Security
  - Access control: We have our own access control where registering users have different level of privileges. First of all,user has to login to view what he/she has uploaded and users can only view what they have uploaded. Law enforcement on the other hand, has privileges to view and download any pde uploaded by any user.
  - 2 factor authentication - Every user has to confirm hi/her email address for him/her to be successfully registered.
  - Hashing: We implement sha256 hash algorithm were we hash users passwords and links to pde before they are send to a cloud.

  - Encryption: We implement AES encryption algorithm. We encrypt users passwords and links to pde before the pde is sent to a cloud.
  - SSL: we use secure layers to connect to the database to ensure extra communication security when sending data and receiving it from the cloud and database.

- Scalability
  - Data storage: We use both the database and the cloud to even up the load. We store users information in the database and user's pde files in the cloud and save the link of those large files in the database. This simplifies database structure and increases performance since the large data files are not queried to/from the database but only the partial link of the file is stored in the database and the file itself is stored in cloud. Every user has a related to a list of links of pde files he/she uploaded on our system.

- Reliability
  - Testing: We are using tested plugins from ngCordova repository to access device's camera, audio and video options. The plugins have been tested and we also do our own integration tests to ensure that the plugins work well with no conflicts.
  - Exception handling: We catch all our exceptions to ensure that the system behaves as intended.
  - Fault tolerance: A copy of the pde file is kept in our system cache until it is successfully uploaded to the cloud, in case there is a connection error while a user tries to upload the pde file. If there is an error, the cached pde file will be resend.

- Integrability
  - Our system is integrated with google map api to decode the longitude and latitude values we get from user's device location to human-readable address.
  - Server: We integrate mysql database with Apache server to save large files on the server and store the link to the files on the database.
  - Plugins: We integrate media plugins to access device's camera, audio and video controls with encryption and hashing algorithms.

- Performance
  - Cloud and Database: We separated user details from large files users upload to increase performance and to structure our data well.
- Usability
  - Ionic framework build hybrid apps that uses native device's controls, meaning users enjoy the feel of using an app that is build with native language on any device the user will be using our system on. The button, forms and checkboxes feel as if they were build from native languages on both android and ios devices, meaning users will ship their knowledge of using the device and making it easier for users to adapt and learn to use the system quicker.
  - Icon: We use visuals such as icons for users who can not even read to be able to still use the system because of the visuals on the buttons.
  - Colors: We use colors to show users the quality and strength of their passwords, making them to enter the right secure password from the start.
- Maintainability
  - MVC: We use MVC design pattern to separate the view from the controller and the model. This helps us modularize our system and make it easy to maintain and test it.
  - State pattern: Every view has it's state and depending on the state, a relevant page loads, this simplifies page linking and makes it easy to load any page dynamically depending on the stage of the current page or on which button pressed.
- Deployability
  - We are building a hybrid system that can be deployed in almost all the platforms. We are focusing on Android, Ios and windows mobile devices because of the higher percentage in the use of these devices. This also reduces the quirks of different devices and ensures reliability on every device and high performance.

2.4 Integration and access channel requirements

The system will be accessed using Mobile/Android Applications clients through the use of ngCordova, browser clients which is where ionic's web based framework comes in and web services clients.

2.5 Architectural constraints

- There are multiple factors that may place constraints on the architecture that is being developed for the uWatch System
- This is developed under ionic framework which is simply web based, thus it put constraints on certain programming languages Compliance with existing standards

3 Architectural pattern or styles

- MVC(Model-View-Controller)
  - Ionic framework is build from this pattern, where the view is the template HTML pages, the model is the JSON object to and from the database and the controller is the JavaScript functions to implement the system's functionality.
- Pipes and Filters
  - We will be tagging images, videos and audio files being uploaded to the server and using this architectural pattern to allow us to easily search and filter by tags and categorize the PDE being uploaded.
- Client/Server
  - uWatch allows multiple clients to access the system using N-tier architectural style.The benefit of using N-tier architectural style is that it improves the scalability of the system.
  - The server side is the back-end of the system which manages the centralized data and access to the database from the server. The aim of having the back-end manages the data is to achieve higher security.

4 Architectural tactics or strategies

- Security: We do not cache user credentials nor keep them in history. This forces users to login everytime they need to login to the our system. Users have a limited amount of tries to login.

- Maintainability and Flexibility: Our views are templated, separating the view from the templates. Since our system relies on AngulaJS, we inject dependencies and as such the is Inversion of control (IoC).

- Perfomance: We catch users' retried PDE so that the system can respond faster especially on slow internet connection. Localstorage caches data in case the relational DB goes down and once it goes up again, it synchronizes the cached data with the data in the DB.

- Scalability: Our server is not tightly coupled with the system. The serve can be hosted any where, locally or remotely. Links to files stored on the server are send to a relation database.

  We use local storage to store any PDE that the user chooses to upload later in case the user did not have internet connection at the time the PDE was captured.

- Reliability and Availability: caching and local storage as explained above.

- Auditability - Users login before they can upload a PDE. - We use date, time-stamps and Geo-location. - Data that is send to and from the serve is encrypted.

- Testing: we will implement unit testing and integration tests to see if the pre and post condition are/were met.

- Usability: We use nice icon and buttons and less text and encourage users who can not read to be able to use our system. Since it is a web base, we develop our system in away that blind and visual impaired users will be able to use screen readers to access our services.

- Integrability: Our system is integratable since it is decoupled form the cloud/database, it can work with any database or a cloud for that matters or simply a local storage.

- Deployability: The system is cross platform, it will be deployable on IOS, Android, Windows, Amazon and blackberry platforms.

5 Use of reference architecture and frameworks

- We are using Ionic framework to implement our mobile application. Ionic framework is a mobile base framework build on the following frameworks:
  - ngCordova An Apache cordova framework which packages HTML, css and js files in to respective platform such as Android, IOS, Blackberry, Windows mobile etc.
  - AngulaJS Is a Google JavaScript framework that helps implement controllers and services for our project.
  - Bower and Node package manager (NPM) to help install dependencies and plugins into our project in order for us to extend functionality beyond your normal JS capabilities.
  - UI-Routes we use routes to changes from one view to the other and from one state to another, this framework help us achieve exactly that.
  - Gulp js for simplifying our project and stripping out all unnecessary comments on our codes so that it can compile faster and respond quicker. Furthermore, the packaged final project file is relatively smaller and making it easy to deploy, share or to store.
  - SASS this is the framework that deals with the look and feel of our application. SASS is described as css on steroid on the sass website.
- Reasons for choosing ionic framework:
  - Since it is a web based, it leverages our web development skills and the knowledge we have of the web based languages such as js, HTML and css.
  - It is cross platform, one project will be deployed on every platform. Meaning we do not need to write special code to accommodate a specific platform or learn the native language that each platform uses. This speeds up production and is convenient since we can not assume that users will be using one specific platform.
  - It is pretty powerful. We use ngCordovas plugins to access native device camera, video camera, audio recording, media file, local storage etc All this is done effortless.

- Our app has to access ones Geo-location and it is relatively easy to that with ionic than will a native language.
- Is easy to implement security features in javascript
- Is easier to debug, you can also debug in a browsers console.
- It uses less resources, you do not need an emulator to test features such as Geo-location or encryption
- There are many encryption js libraries. For example forgejs, cryptojs and they both implement AES cryptography.
- It's easy to work with JSON object.
- It has a great community and up to date plugins.

6 Access and integration channels

- Integration channels that we using:
  - We using a REST API which uses the Library cURL(which is a computer software project providing a library and command-line tool for transferring data using various protocols)
  - This REST API uses :
    - HTTPS (Hypertext Transfer Protocol Secure) is a protocol that allows safe and secure transfer of data over a network.
    - HTTP PUT for retrieving, writing and updating of data.
  - API specifications used in the integration of the systems
    - Authentication API - We have our own access control were users are required to verify their email address by OTP
    - Security API - which an API that specifies rules(read,write and validate),what the client can be able to access and what restricted to them and it provides encryption strategies,for example Forge(which is an encryption method).
  - Quality requirements that can be achieved through the implementation of the access channels mentioned.
    - Reliability so that the system is online as much as possible and that data transfer isnt easily corruptible and that the transfer is fast. We use localstorage to back up the file before it is send to the server, if the file transfer fails or the file gets corrupted, then the local copy is tried until the file is successfully uploaded.

    * Security of user data being transferred.
    * Scalable so that a large amount of users data can be transferred to and from the systems.
    * Maintainable, the integration with the system must be easily maintained. (because for example security rules are all defined in one place.

7 Technologies

Technologies Used in the development of our Application are stated below along with why the choice was made and it's benefits.

- Ionic framework
    * Ionic framework is a great framework that supports hybrid application which has many benefits, specifically in terms of access to third party code , speed development and platform support.
    * It is an Model-View-Controller framework which separates concerns, allowing re-use of the business logic; it enables parallel development by separate teams which is ideal when working in a project such as this one. The MVC framework uses AngularJS for controllers and services, HTML for view and JSON for the model which are all popular and easy to learn languages.
- ngCordova
    * ngCordova is a collection of 63+ AngularJS extension on top of the cordova API that make it easy to build, test and deploy Cordova applications with AngularJS, ngCordova uses packages and libraries, compiles source code and deploys the source code to Apk (android), Xap (windows mobile) and ipa (ios).
- Security provided by the serve includes:

- Authentication API-who the user is.
- Support SSL on all clients
- Uses Bcrypt for password storage
- Uses the JSON Web Token for standard credentials

- Generate Server-Signed Tokens
- All security Logic is put in one pace

The benefits of using PHP and MYSQL:

- PHP has many libraries for encryption and hashing
- Is easier to send email with php than with many server side language
- PHP works well with session and tokens
- MYSQL is easy to work with
- Is easier to manage user details and writing sql queries

8 References

- All About Ionic (author and date unknown) Available from:http://ionicframework.com/docs [Accessed: 31 july 2015]
- Why use ngcordova (author and date unknown) Available from:http://stackoverflow.com/qu [Accessed: 31 july 2015]
- Features (author and date unknown) [Accessed: 31 july 2015]