COS 301 HighTech Team - SRS Document

Theoveshan Naidu
16148861

Janaki Patil
16006110

Zi Xin Zhang
15192556

Alexandros Petrou
15291792

Tristan Sander-hughes
17071390

2 May 2019

Contents

1 Introduction

In the modern plumbing industry, a continuous source of inconvenience arises for plumbers to log information on the geysers and components installed on various jobs by means of written material on a day-to-day basis. Considering this the goal of this project is to design a new database application system that greatly reduces the inconvenience of having to log information about installed geysers by taking advantage of the prevalence of mobile devices in the modern world. In order to do this, we will shift the logging of details from the paper-based method itself to a mobile application, detailed below.

1.1 Definitions,Acronyms

This section provides definitions of all items, acronyms and abbreviations to interpret the SRS document properly.

1. Certificate of Compliance - A certificate to prove that the geyser installation complies with regulation standards.
2. App - Application

2 User Characteristics

2.1 Intended users

The are different kinds of users that will be interacting with our system. The intended users of the software are:
Plumber- Geyser Installers
The most interactive part of the app consists of what the installer will be doing. The installer would be a user that is installing the geyser and associated parts, they require a fast and easy way to compile all related information files and images onto a database in order to streamline future adjustments and insurance claims.
Insurance Companies
Insurance companies will be able to be easily contacted in case of a claim and appropriate documents can be easily accessed or sent to the linked insurance companies. Insurance users will no longer have to find the documents needed to make a claim as these are stored online. Should a problem arise the insurance company or those in contact with them can be notified of a possible claim.
Homeowners
Homeowner characteristics include the need for streamlining between manufacturer of the geyser, insurance companies, and plumbers. This also includes ease of access to geyser information when combined with easily ac-

cessed plumbers who can advise if the homeowner seeks to find an issue with the geyser or repair it.

3 Technology and Constraint

3.1 Technology Decisions

3.1.1 Ionic Framework

We chose Ionic Framework because is a free open source mobile development framework that allows us to create cross-platform applications that are native to the platform. Ionic allows us to create applications with Javascript/Typescript, HTML and CSS. We wanted that functionality because it would allow us to not only port our application into a web-app, but also a desktop environment using Electron Framework all using a single codebase.

3.1.2 Typescript

Typescript is a typed superset of Javascript that compiles into plain Javascript. We choose this language because it integrates well with Ionic, and it allows us to write typed code which helps us catch errors at compile time, and also helps us debug our application much easier than a dynamic language. Typescript is also free and open source with a large community that would help us get libraries and help that we might need.

3.1.3 Github

We chose to use Github because it is a free centralized area what we can all collaborate using the Git Work-Flow that allows us all to work on the same project effectively and with as less conflict as possible. It has many different tools that allows us to review each others work, and also help.

3.2 Constraints

3.2.1 Android API levels

We are limited to only working with Android devices and because android devices each all have different API levels that allow for different features and functionality, we had to make sure that our application was limited to only using the API level of the lowest common android.

3.2.2 Open Source / Free

We are limited in only using open source or free software and framework to build our application, propriety software would be too expensive, or a limitation should this application wish to do something that is against their software policies.

4 Function Requirements

4.1 Use-case

UC1 The plumber login with its credentials.
UC2 The plumber scans the barcode that is placed on the geyser.
UC3 The plumber can view the heap map.
UC4 The plumber uploads hot water system images.
UC5 The plumber uploads images of the house where the geyser is being installed.
UC6 The plumber enters geysers details.
UC7 The plumber enters his/her details.
UC8 The plumber enters the home owners details.
UC9 The plumber enters the insurance companies' details.
UC10 The plumber uploads the Certificate of Compliance.
UC11 The plumber updates the details if repairs are done on the geyser.
UC12 The plumber can view a report (summary of the details entered).
UC13 The home owner scans the barcode that's placed on the geyser.
UC14 The home owner enters his/her credentials.
UC15 The home owner can view the report of the installation of geyser.
UC16 The home owner sends an email notification regarding the claim on geyser to the insurance company.
UC17 The home owner receives a notification upon successfully sending an email to the insurance company.
UC18 The home owner sends a query to the manufacturer of the geyser.
UC19 The home owner receives a notification upon successfully sending a query to the manufacturer of the geyser.
UC20 The home owner sends a query to the plumber.
UC21 The home owner receives a notification upon successfully sending a query to plumber.
UC22 The administrator receives push notification when updates are made by the installer.
UC23 The administrator views the installation report.
UC24 The insurance company login with their credentials in order to proceed.

4.2 Functional requirements

Authentication subsystem

1 The app shall provide authentication functionality.

1.1 The app should allow the users to scan barcode on the geyser in order to proceed.

1.2 The app should differentiate between the different types of users (installer, admin, land owner or insurance company) and only return data relevant to that specific type of user.

Data Capturing subsystem

2 The app shall allow various details to be captured after the installation of the geyser.

2.1 The app must allow the plumber to enter his/her details.

2.2 The app must allow the plumber to enter the Geyser Details

2.3 The app must allow the plumber to upload installation related images.

2.3.1 The app must allow the plumber to upload Hot water system images.

2.3.2 The app must allow the plumber to upload Certificate of Compliance.

2.3.3 The app must allow the plumber to upload images of the house where the installation of geyser was done.

2.4 The app must allow the plumber to enter Insurance company details.

2.5 The app must allow the plumber to enter home owner details.

Report subsystem

3 The app shall generate a report for various users.

3.1 The app should generate a report based on the details captured for the administrator.

3.2 The app should generate a report based on the details captured for the home owner.

3.3 The app should generate a heat map of geysers based on different criteria.

Notification Subsystem

4 The app shall support various notification functionality.

4.1 app should notify the admin if changes are made to the database.

4.2 The App should allow the Residence owner to send notification

4.2.1 The app should allow the Residence owner to email the insurance company regarding their claim

4.2.2 The app should allow the residence owner to send query to the manufacturer of the geyser

4.2.3 The app should allow the residence owner to send query to the plumber who installed the geyser.

5 Traceability Matrix

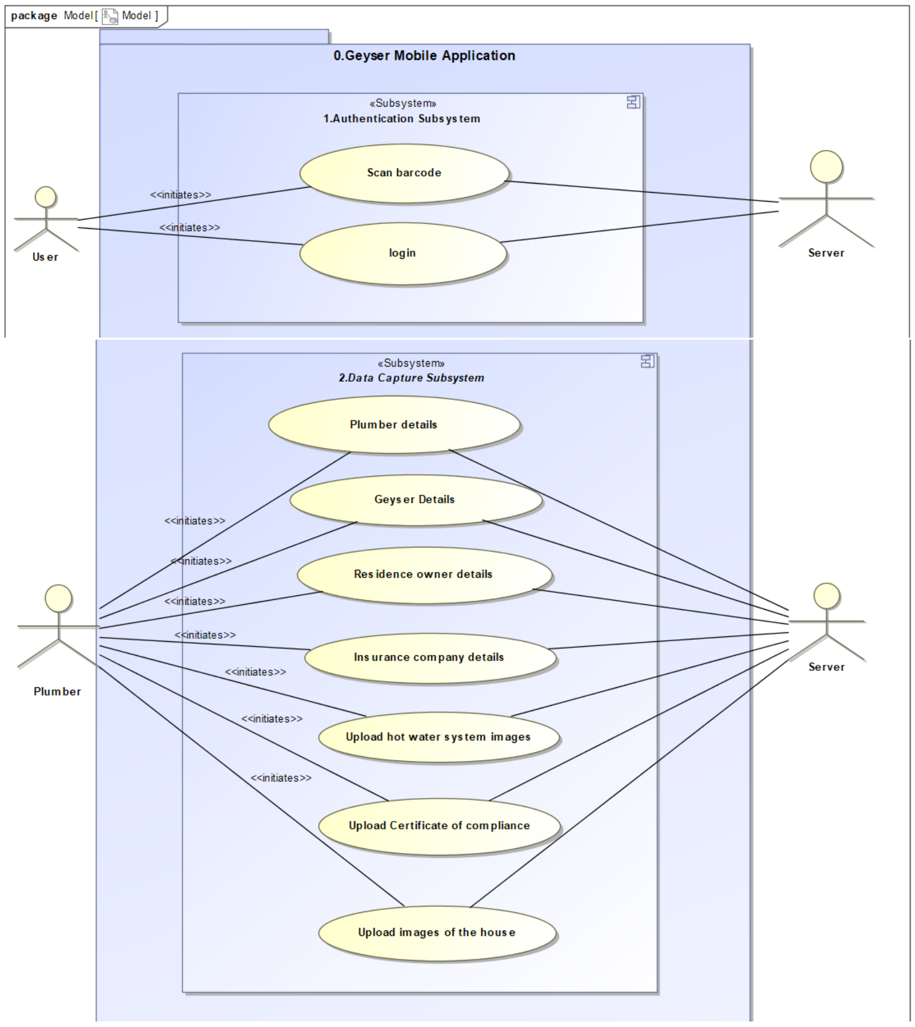
	Authentication	Data Capture	Report	Notification
Functional				
1.1	X	X		X
1.2	X			X
2.1		X		
2.2		X		
2.3		X		
2.3.1		X		
2.3.2		X		
2.3.3		X		
2.4		X		
2.5		X		
3.1			X	
3.2			X	
3.3			X	
4.1				X
4.2		X		X
4.2.1				X
4.2.2				X
4.2.3				X
Quality				
QR1		X	X	
QR2		X		
QR3	X			X
QR4	X	X	X	
QR5		X	X	
QR6	X	X		
QR7			X	
QR8		X		

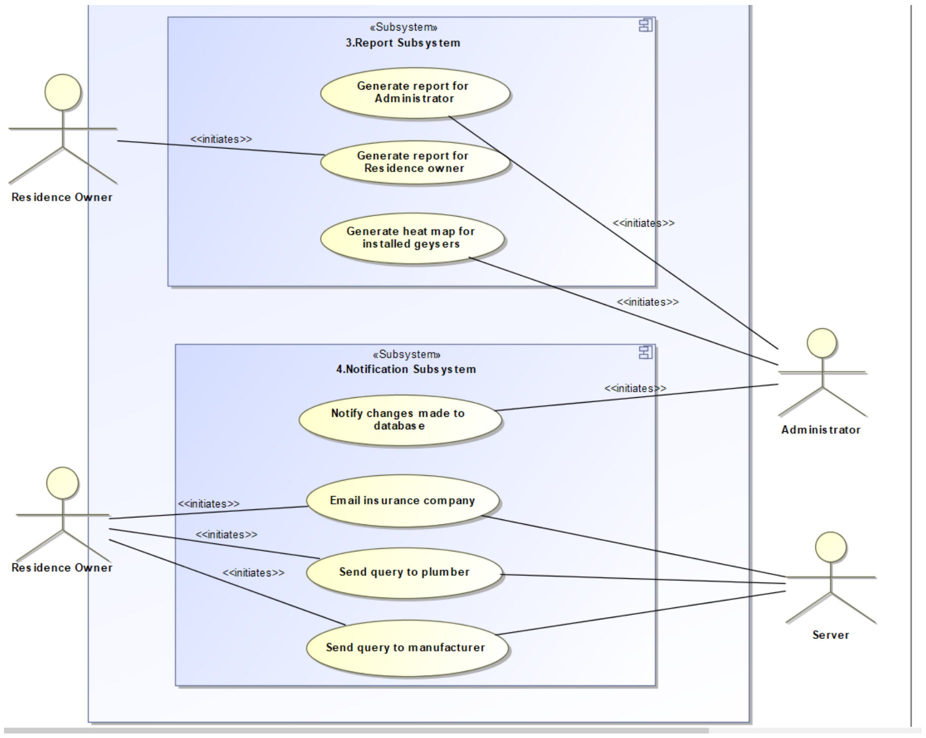
6 Quality Requirements

QR1: Performance
(a) - The app shall check how many geysers are installed a day. This allows the app to update the heat map with additions of new geysers.
(b) -The app should upload the information to the database within 20 seconds of being connected to the internet.
QR2: Reliability
(a) -The information will be stored locally on the phone until it is uploaded to the database on the server where it will have the ability to be accessed.
(b) -The database shall keep all information of every geyser(CRUD), in case of a server failure, all the information should also be duplicated to another storage area for redundancy.
QR3:Security
(a) -Since information stored on the database needs to be confidential and comply with the popi act the database needs to be secured by password access to ensure security.
(b) -Incorrect credentials entered need to alert the admin(by sending notification) that someone is trying to access the application without permission.
(c) -Access permissions for particular system information shall only be changed by the system administrator.
QR4: Scalability
(a) -The app server must be able to scale the amount of computing power it uses depending on the amount of devices currently connected, so it must be able to make more processing power available if the number of connected users rise above a certain.
QR5: Availability
(a) -The app shall allow users the enter data from any location and upload whenever they have internet signal
(b) -The app shall show users the locations of geysers with the corresponding filter options on a heatmap to show certain geysers of various attributes for example manufacturer of the geyser, model,size etc.
(c) -The system shall be available for 99 percent of the time.
QR6: Maintainability
(a)- The database will allow an admin user to maintain all information on the database incase changes need to made.
(b)- The app will be created in a modular fashion to allow changes to be made to modules without affecting the whole app.
QR7: Monitorability
(a) -The app will make use of multiple heat map views to display different details of geysers in the area.
QR8: Usability
(a) -App users who want to enter information or edit information shall be able quickly and easily to to accomplish any task. -the app shall process actions in less than 30 seconds

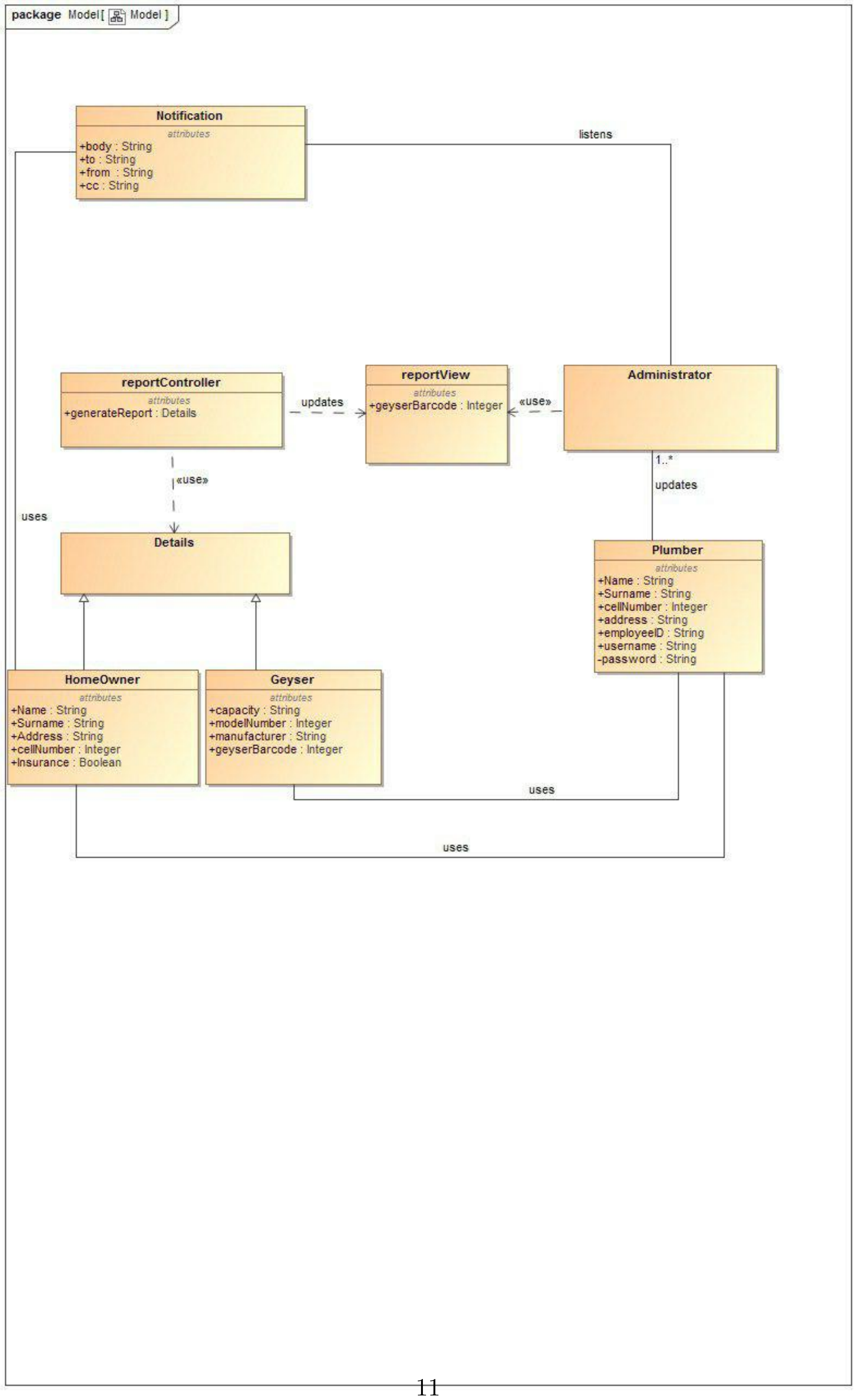
7 Diagrams

7.1 UML use case diagrams

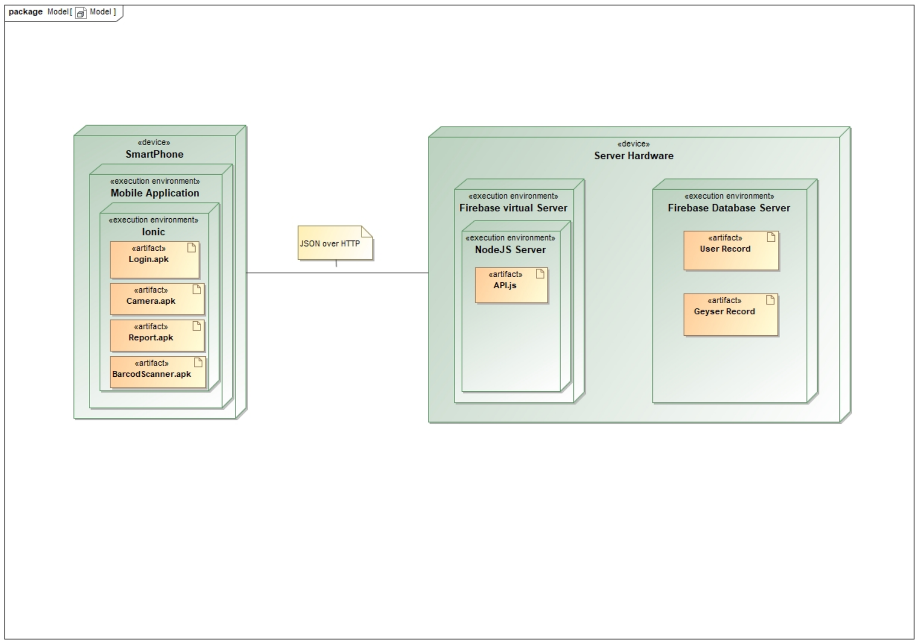



«Subsystem»
3.Report Subsystem
Generate report for Administrator
Generate report for Residence owner
Generate heat map for installed geysers
<<initiates>>
<<initiates>>
<<initiates>>
Residence Owner
Administrator
«Subsystem»
4.Notification Subsystem
Notify changes made to database
Email insurance company
Send query to plumber
Send query to manufacturer
<<initiates>>
<<initiates>>
<<initiates>>
<<initiates>>
Residence Owner
Server

7.2 Domain Model



7.3 Deployment Diagram



8 Architectural Design

Architectural requirements and justification Our overall architecture is a mix of multiple different architectural styles in order to ensure that our system can allow for separation of concerns, loose coupling, and levels of abstraction. To begin it makes use of a database-centric architecture, through the integration of a centralized database which helps facilitate data storage and sharing between data accessors. These data accessors all use the database as a means of indirectly talking with each other. This type of architecture reduces the overhead for data transfer between data accessors. This also provides the system with the ability to be scaled up easily. The next architectural style that is in use in our system is a client server architecture, which exists between the device (Android Smartphone, tablets) the web server and the app server. This type of architecture limits the client server relationship to a request-response messaging pattern thereby allowing the system to be easily scalable, such as by adding more clients. This type of architectural style could also be considered N-Tier because of the use of an application server, which is a client of the database. This design is also used because it makes provisions for security concerns. The server uses an n-layer architecture with a persistence architecture as its final layer (MVC pattern with a REST API as the view). The n-layer architecture at its first level is a REST API, and as such is an interface of the functions available to the clients. This API communicates with the next layers, controlled using Node.js , which control business logic. The final layer is a persistence layer consisting of an interface to a cloud firebase database. By using a persistence architecture here, the admin can change database(such that can perform CRUD functionality) without modifying the entire system (low coupling is achieved). The smartphone application uses an n-layer architecture, as is good for interface design. This is a suitable architecture, as the application is designed using Angular and Ionic, both of which utilize the MVC (Model View Controller) pattern. The first layer is the View, presenting the user with a user interface, rendered to the user using JavaScript, HTML and CSS, the second is the Controller, implemented in client-side JavaScript, which controls client side business logic (any processing which can be offloaded from the server, such as aggregating data), and the last layer is the Model. The model layer is simply an HTTP interface (Angular HttpClient/Browser fetch API) to the server. Using the n-layer architecture with MVC provides low coupling and high cohesion; modification to the user interface of the application can be done without having to modify business logic, and features can be added, removed or altered with ease. This means designers and programmers can work concurrently, and testing of new interfaces (e.g. A/B testing) can be performed without modifying functionality.

9 Conclusion

The new way of loading geyser installation details need to comply with the current direction in which reporting is done all over the country - into the mobile, portable, personal sphere. With the system specified in this document, the same old logging functionality is shifted into this new sphere - as well as gaining some improvements.
In conclusion the proposed system will change the way plumbers interact with the logging of information. More specifically it will focus on evolving the way plumbers and homeowners submit their information by moving towards a more mobile approach allowing users to upload images and type out legible data. Interfaces will be created on Geyser24(mobile app) and will speed up the process of uploading of info and allowing a user to simply enter their geyser number and display the generated report with all major details included.
In turn the process times will greatly be reduced as well as increasing security by encrypting important data. Geyser24 will also allow users to log queries about their geyser to receive help faster. Geyser24 will change the plumbing industry as well as the household as it will provide the fastest and easiest way to log issues and keep track of geyser information.

10 References

Adams, D. (1995). *The Hitchhiker's Guide to the Galaxy.*

Groenfeldt, T. (n.d.). *Some Banks Are Heading To The Cloud -- More Are Planning T.* Retrieved from Forbes: https://www.forbes.com/sites/tomgroenfeldt/2014/06/26/some-banks-are-heading-to-the-cloud-more-are-planning-to/#75e6fa7158c9

Schwartz, B. (n.d.). *Do the Math: Can Your Business Benefit From a Cloud Database?* Retrieved from Tech: https://tech.co/news/math-can-business-benefit-cloud-database-2015-08

Zamfiroiu, A. (2012). *Integrability and Interoperability of Mobile*. Retrieved from Zamfiroiu: https://www.researchgate.net/publication/236648829_Integrability_and_Interoperability_of_Mobile_Applications