**Requirements Specification (RS)**

GROUP H

|  |  |  |
| --- | --- | --- |
| Penuel Maypa | x16382003 | BSHC |
| Rehan Naeem | x16333223 | BSHC |
| Jodeyne Teneza | x16408022 | BSHC |
| Lee Sharidan | x16353613 | HCC |
| Alpheus Kakkattupara | x16120078 | HCBC |

|  |  |
| --- | --- |
| Admin | Penuel Maypa |
| Project Manager | Penuel Maypa |

|  |  |
| --- | --- |
| Document Started: | 01/Feb/2017 |
| Document Finished | 16/Feb/2017 |

Requirements Specification (RS)

Document Control

Revision History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date** | **Version** | **Scope of Activity** | **Prepared** | **Reviewed** | **Approved** |
| 02/02/18 | 1 | Create Introduction | Penuel | X | X |
| 02/02/18 | 3 | Create Purpose | Penuel | x | x |
| 02/02/18 | 3 | Created Project Scope | Penuel | x | x |
| 02/02/18 | 3 | Created Definitions, Acronyms, and Abbreviations | Penuel | x | x |
| 02/02/18 | 3 | Mock-Ups | Jodeyne | x | x |
| 07/02/18 | 4 | Non- Functional | Lee | X | X |
| 07/02/18 | 4 | Description Priority | Lee | x | x |
| 07/02/18 | 4 | Performance/Response | Lee | x | x |
| 07/02/18 | 4 | Availability Requirement | Lee | x | x |
| 07/02/18 | 4 | Recover Requirement | Lee | X | X |
| 07/02/18 | 4 | Robustness Requirement | Lee | x | x |
| 07/02/18 | 4 | Functionalities Requirement | Rehan | x | x |
| 09/02/18 | 4 | System Architect | Penuel | x | x |
| 09/02/18 | 4 | ER Diagram | Penuel | X | X |
| 09/02/18 | 4 | Use Case Document | Rehan | x | x |
| 09/02/18 | 4 | Use Case Document | Jodeyn | x | x |
| 12/02/18 | 5 | Data Model | Penuel | x | x |
| 12/02/18 | 5 | Class Diagram | Penuel | X | X |
| 13/02/18 | 5 | Requirement 1 | Lee | x | x |
| 13/02/18 | 6 | Performance/Response | Lee | x | x |
| 13/02/18 | 6 | Availability Requirement | Lee | x | x |
| 13/02/18 | 7 | Recover requirement | Lee | X | X |
| 13/02/18 | 7 | Security requirement | Lee | x | x |
| 13/02/18 | 7 | Reliability | Alpheus | x | x |
| 13/02/18 | 7 | Maintainability | Alpheus | x | x |
| 13/02/18 | 7 | Portability | Alpheus | X | X |
| 13/02/18 | 7 | Extendibility | Alpheus | x | x |
| 13/02/18 | 9 | Requirement - Children Login | Jodeyne | x | x |
| 13/02/18 | 9 | Requirement - Create Chore | Jodeyne | x | x |
| 13/02/18 | 9 | Requirement - Log In | Jodeyne | X | X |
| 13/02/18 | 9 | Requirement - Set Reward | Jodeyne | x | x |
| 13/02/18 | 9 | Requirement - Earn XP | Jodeyne | x | x |
| 14/02/18 | 10 | Requirement - Assign Chore | Rehan | x | x |
| 14/02/18 | 10 | Requirement - Confirm Chore | Rehan | X | X |
| 14/02/18 | 10 | Requirement - Remind User | Rehan | x | x |
| 14/02/18 | 10 | Requirement - Send Chore | Rehan | x | x |
| 15/02/18 | 11 | UML Diagram - Main System | Penuel | x | x |
| 15/02/18 | 11 | Use Case - Synchronise Data | Penuel | x | x |
| 15/02/18 | 11 | Use Case - Create Account | Penuel | x | x |
| 15/02/18 | 11 | GUI | Jodeyne | x | x |
| 16/02/18 | 12 | System Evolution | Penuel | X | x |

Distribution List

|  |  |  |
| --- | --- | --- |
| **Penuel** | **Jodeyne & Rehan** | **Lee & Alpheus** |
| Table of Contents   * 1 Introduction * 1.1 Purpose * 1.2 Project Scope * 1.3 Definitions, Acronyms, and Abbreviations * System Architecture * System evolution | * 2 User Requirements Definition * 3 Requirements Specification   + 3.1 Functional requirements     - 3.1.1 Use Case Diagram     - 3.1.2 Requirement 1   <name of requirement in a few words>   * + - 3.1.3 Requirement 2   <name of requirement in a few words>   * GUI | * 3.2 Non-Functional Requirements   + Performance/Response time requirement   + Availability requirement   + Recover requirement   + Robustness requirement   + Security requirement   + Reliability requirement   + Maintainability requirement   + Portability requirement   + Extendibility requirement   + Reusability requirement   + Resource utilization requirement |

**Table of Contents**

[Requirements Specification (RS) 1](#_Toc9798)

[Document Control 1](#_Toc22604)

[Revision History 1](#_Toc28075)

[Distribution List 3](#_Toc12777)

[1 Introduction 6](#_Toc1038)

[1.1 Purpose 6](#_Toc27948)

[1.2 Project Scope 6](#_Toc7839)

[1.2.1 Scope 6](#_Toc24979)

[1.3 Definitions, Acronyms, and Abbreviations 8](#_Toc31169)

[2 User Requirements Definition 9](#_Toc6476)

[3 Requirements Specification 10](#_Toc8412)

[Functional requirements 10](#_Toc15431)

[3.1.1 Use Case Diagram 11](#_Toc26017)

[3.1.2 Requirement 1: Create Account 16](#_Toc238)

[3.1.3 Requirement 2: Log-In 17](#_Toc2666)

[3.1.4 Requirement 3: Children Log-In 19](#_Toc14206)

[3.1.5 Requirement 4: Create Chores 21](#_Toc157)

[3.1.6 Requirement 5: Assign Chore 22](#_Toc27994)

[3.1.7 Requirement 6: Set Reward 23](#_Toc25412)

[3.1.8 Requirement 7: Remind User 24](#_Toc20435)

[3.1.9 Requirement 8: Send Chores 25](#_Toc22320)

[3.1.10 Requirement 9: Confirm Chore 26](#_Toc29230)

[3.1.11 Requirement 10: Earn XP 27](#_Toc1375)

[3.1.12 Requirement 11: Synchronise Data 28](#_Toc30972)

[Non-Functional Requirements 29](#_Toc18823)

[3.1.13 Performance/Response time requirement 29](#_Toc31990)

[3.1.14 Availability requirement 29](#_Toc15849)

[3.1.15 Recover requirement 29](#_Toc18942)

[3.1.16 Security requirement 29](#_Toc22242)

[3.1.17 Reliability requirement 29](#_Toc764)

[3.1.18 Maintainability requirement 30](#_Toc5070)

[3.1.19 Portability requirement 30](#_Toc14753)

[3.1.20 Extendibility requirement 30](#_Toc14396)

[4 GUI 31](#_Toc16738)

[5 System Architecture 37](#_Toc22401)

[5.1 Database -- Data Model 38](#_Toc32546)

[5.2 Class Diagram 41](#_Toc3630)

[6 System Evolution 42](#_Toc17261)

[7 References & Bibliography 43](#_Toc24569)

# Introduction

# Purpose

The purpose of this document is to set out the requirements for the development of Mom Says. This document contained an in-depth detail of the app. It will explain how the app will work, its system, its back-end, its features and its architecture.

It will also explain system constraints, interface and interactions with other external applications.

# Project Scope

The scope of the project is to develop an app that would let parents manage chores and assigned the chores to their children.

### Scope

**WEEK 3**

* Define the Requirement Specification of this project.
  + Define its Use Case
  + Define its users
* Development tools and API Research:
  + Research the best programming language for the App.
  + Research and find out the best IDEs and tools to use for the development of the application.
  + Research and find out what API and frameworks we need to implement.
  + Research the most suitable type of database management system to manage the back-end and manage data.

**WEEK 4 - 8**

* PART 1 - Development and Implementation
  + GUI and User Experience
    - Design the GUI for the mobile App
    - Design the GUI for desktop
  + Coding/Development
    - Code and develop each functionalities
    - Implement API’s and framework
    - Develop the database

**3 DAYS BEFORE MIDPOINT PRESENTATION**

* PART 2 - Development and Implementation
  + GUI and User Experience
    - Design the GUI for the mobile App
    - Design the GUI for desktop
  + Coding/Development
    - Code and develop each functionalities
    - Implement API’s and framework
    - Develop the database
* **MID POINT TESTING - Prototype Testing**
  + Unit Testing
  + Integration Testing
  + System Testing
  + Operational Testing

**WEEK 9 - 12**

* FINAL TESTING - Testing and Pre-release
  + Unit Testing
  + Integration Testing
  + System Testing
  + Operational Testing
* **Final Documentation**
  + Project Retrospective
  + Refurbish final Documentation
  + Prepare Presentation for Demonstration

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **ACRONYMS & ABBREVIATIONS** | **DEFINITIONS** |
|  |  |
| API | Application Programming Interface |
| GUI | Graphical User Interface |
| IDE | Integrated development environment |
| JSON | JavaScript Object Notation |
| NoSQL | NoSQL, which stand for "not only [SQL](http://searchsqlserver.techtarget.com/definition/SQL)," is an alternative to traditional relational databases in which data is placed in tables and data [schema](http://searchsqlserver.techtarget.com/definition/schema) is carefully designed before the database is built. |

# User Requirements Definition

* This is a chore app are for parents.
* Its function is to remind children of their chores.
* You are able to set notifications with the ability to automatically send reminders via a messaging service app ex. WhatsApp or Facebook Messenger.
* The notification can only be stopped by the parent when the chore is completed.
* There is a reward system is based off of an xp system typically used in games.
* By completing chores, you will get xp. Enough xp is equal to 1 star. A star can be turned in for a reward of the parents liking.
* Stars act as a currency so the more stars the child has the better the reward they can get. This should encourage them to do more chores to get better rewards.
* The parents will be given pre-set chores when they download the app.
* They can send these chores with the custom tag “MOM says” or DAD says”. This can be changed in the settings menu.
* They can also create their own chores and save them to be used for another time.
* The parent can set a daily/weekly/monthly reminder for one of their children.

# Requirements Specification

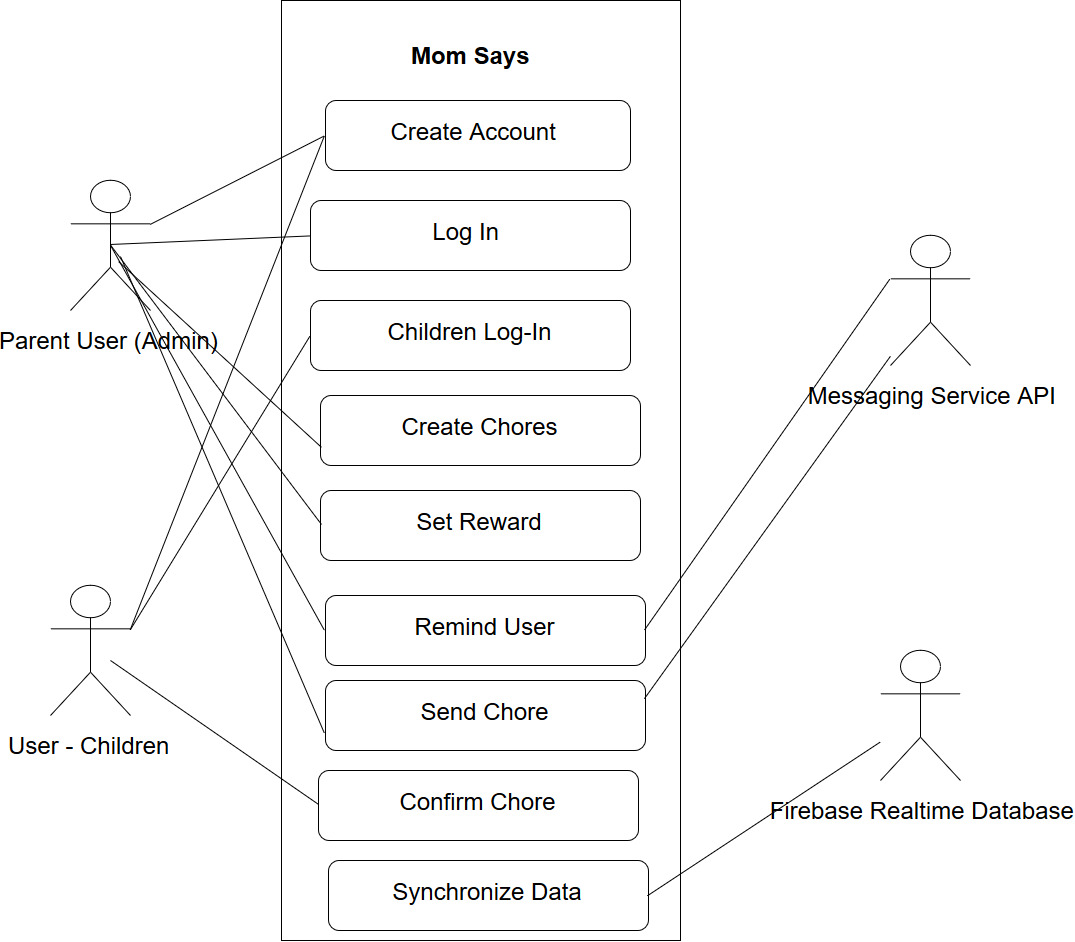
## Functional requirements

|  |  |
| --- | --- |
| Create Account | Users must be able to create an account according to the type of users:   * Parents (Admin) * Children |
| Log In | Users must be able to log-in to their accounts |
| Create Chores | Parents (User Admin) must be able to create chores |
| Set Reward | Parents must be able to Set Reward for each task |
| Remind User | Children (User) must be reminded about their chores. |
| Send Chore | Parents must be able to also send reminders of the chore manually or automatically via Messaging Services App. |
| Confirm Chore | Children must be able to confirm the chore once they have achieve the chore assigned to them by the parents. |
| Synchronize Data | All data from all devices must be synchronize with the use of cloud database. |

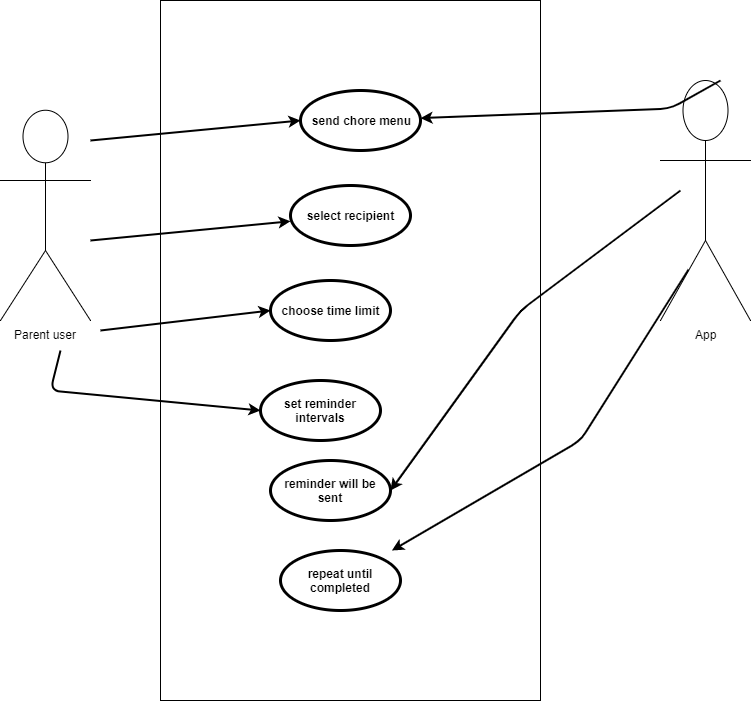
### Use Case Diagram

The Use Case Diagram provides an overview of all functional requirements.

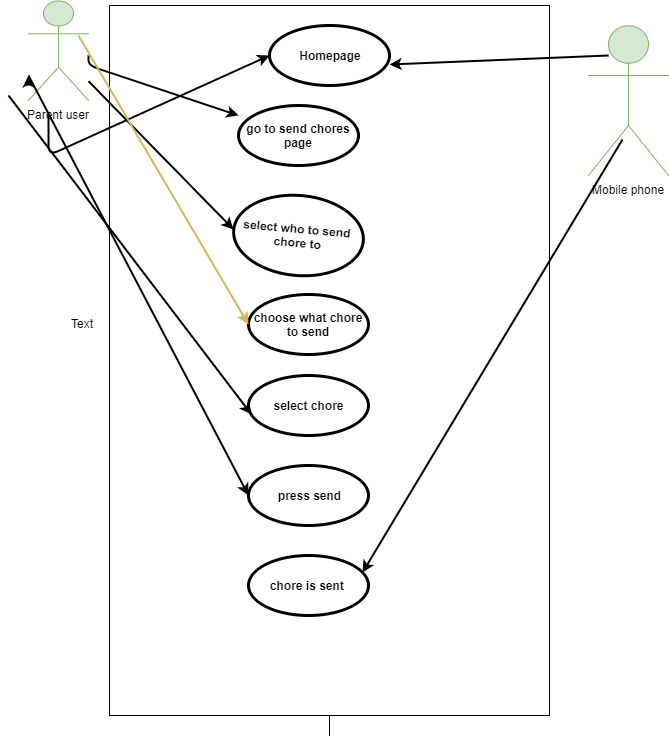
#### Overall System (Main System)



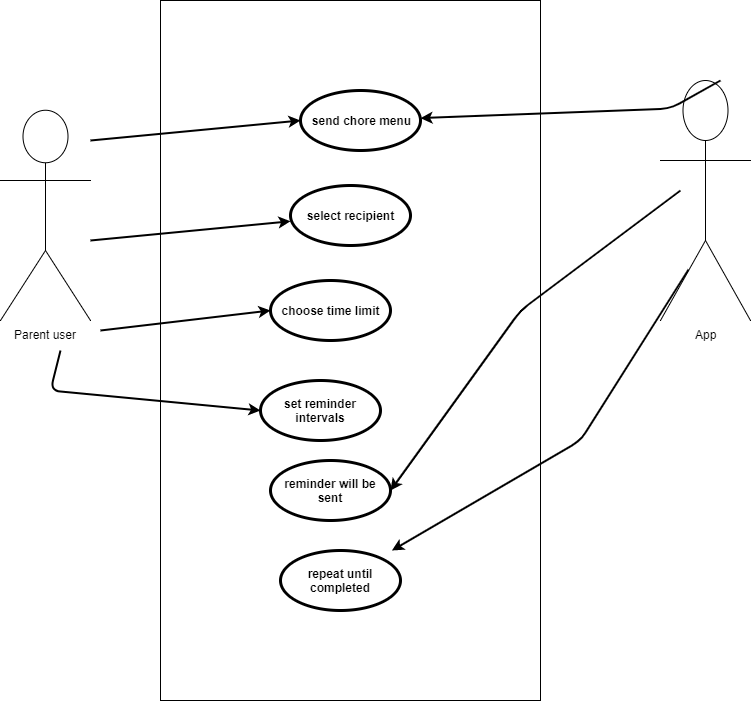
#### Sub System : Assign Chore



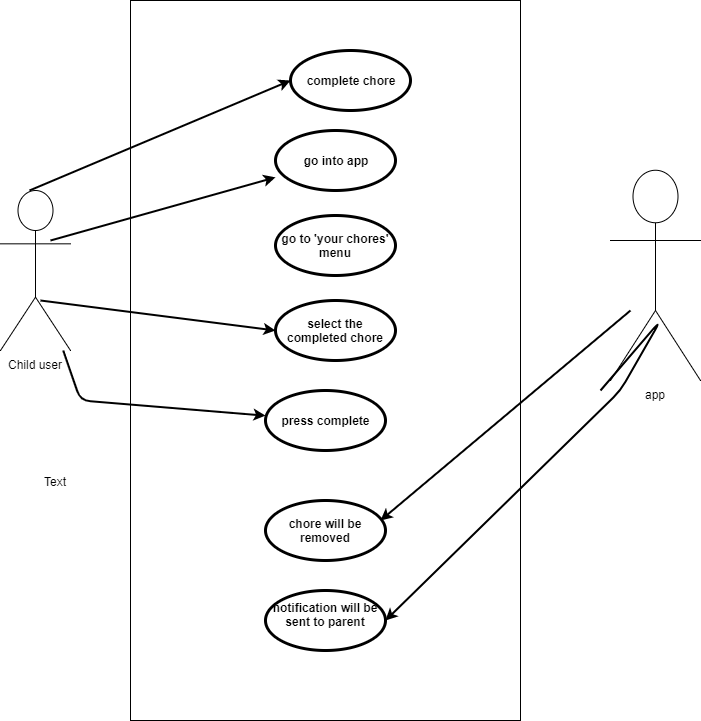
#### Sub System : Send Chore



#### Sub System : Remind Chore



#### Sub System : Confirm Chore



### Requirement 1: Create Account

#### Use case

Create Account

#### Scope

The scope of this use case is for user to create an account by Signing up.

#### Description

This use case describes the creation of a user’s account before they use the app.

#### Flow Description

**Precondition**

Start the app

**Activation**

This use case starts when a user select the “Sign -Up” button.

**Main flow**

1. User selects the “Sign Up” button
2. Fill out the form
3. Select type of user (A1)
4. Select “Create Account” button when finish.

**Alternative Flow**

A1 - Select the following user type:

* Parent (Admin)
* Children

**Exceptional flow**

Displays warning message when forms are not completely filled.

**Termination**

The system stores the user account information

**Post condition**

Brings back to the homepage to Log In.

### Requirement 2: Log-In

#### Use Case

Log In

#### Scope

The scope of this use case is for a user to Log in to their account.

#### Description

This use case describes the Log In procedure for the customer

#### Flow Description

**Precondition**

The system is idle.

#### Activation

This use case starts when a user opens the app for the first time.

#### Main flow

1. User opens the app
2. User presses log in button
3. Prompt will show to ask user to enter log in details
4. User enters details
5. User presses log in

#### Alternative flow

(A1)

1. The system asks the user to make an account
2. The user makes an account
3. Continue to main flow 4

(A2 Connect Google Account)

1. The user signs up using their Google account
2. The system will show a prompt for the permissions
3. The user will either agree or disagree to prompt
4. Continue to main flow 4

(A3 Connect Facebook Account)

1. The user signs up using their Facebook account
2. The system will show a prompt for the permissions
3. The user will either agree or disagree to prompt
4. Continue to main flow 4

Exceptional flow

-

Termination

The system stores the sign in information

Post condition

The system allows the user to the main page

### Requirement 3: Children Log-In

#### Use case

Log In for children

#### Scope

The scope of this use case is for a children to Log in to their account.

#### Description

This use case describes the Log in procedure for the children Flow Description

#### Precondition

The system is in idle.

#### Activation

This use case starts when a user opens the app for the first time.

#### Main flow

1. User gets code from parents’ account
2. User opens the app
3. User presses log in button
4. Prompt will show to ask user to enter log in details
5. User enters details
6. User enters family Code
7. User presses log in

#### Alternative flow

(A1)

1. The system asks the user to make an account
2. The user makes an account
3. Continue to main flow 4

(A2 Connect Google Account)

1. The user signs up using their Google account
2. The system will show a prompt for the permissions
3. The user will either agree or disagree to prompt
4. Continue to main flow 4

(A3 Connect Facebook Account)

1. The user signs up using their Facebook account
2. The system will show a prompt for the permissions
3. The user will either agree or disagree to prompt
4. Continue to main flow 4

#### Exceptional flow

-

#### Termination

The system stores the sign in information.

#### Post condition

The system allows the user to the main page.

### Requirement 4: Create Chores

#### Use case

Create a Chore

#### Scope

The scope of this use case is for a user to create a Chore

#### Description

This use case describes the Chore creation process of the app

#### Flow Description

**Precondition**

The user logs in to the app.

**Activation**

This use case starts when a user presses the chores button on the main page.

**Main flow**

1. User opens the app

2. User presses the chores button

User presses “Create Chore

Alternate flow

-

Exceptional flow

-

**Termination**

The system stores the sign in information.

**Post condition**

The system allows the user to the main page.

### Requirement 5: Assign Chore

#### Use case

Assign Chore

#### Scope

The scope of this use case is to assign the user specific chores

#### Description

This use case describes the process of assigning specific chores to child users.

#### Flow Description

Precondition

The system is in idle.

Activation

This use case starts when the parent is on ‘give chores’ menu.

Main flow

1. The parent user goes to the ‘assigned chores’ section of that menu.
2. The user can assign specific chores to specific people by pressing the ‘+’ button
3. Once they have saved the chore the parent user can set if the chore will be sent automatically or manually.
4. The parent user will then be given the option to ‘give chore’.

Alternative flow

-

Exceptional flow

-

Termination

The system stores assigned chores.

Post condition

The system sends the chore.

### Requirement 6: Set Reward

#### Use case

Set Reward

#### Scope

The scope of this use case is for a user to set the reward they want to give to their children.

#### Description

This use case describes the reward setting for the app.

#### Flow Description

#### Precondition

The user logs in to the app

#### Activation

This use case starts when a user goes to the settings press the reward menu.

#### Main flow

1. User opens the app
2. User presses log in button
3. User goes to settings
4. User presses reward menu
5. User changes the default rewards
6. User presses save

#### Alternative flow

-

**Exceptional flow**

-

**Termination**

The system stores the new Rewards.

**Post condition**

The system returns the user to the settings menu.

### Requirement 7: Remind User

#### Use case

Reminder.

#### Scope

The scope of this is for the child user to be reminded of the chores that haven’t been completed.

#### Description

This use case describes the how the child user will be reminded of the chore they have to complete.

#### Flow Description

Precondition

The system is idle.

Activation

This use case the chore has been sent to the child user.

Main flow

1. The child user has a set time limit of when the chore should be completed by the parent user.
2. The parent user sets intervals in which the reminder should be sent.
3. The app will then send notification via third party messaging app (ex. WhatsApp).
4. This will repeat until the chore is completed.

Alternative flow

-

Exceptional flow

-

Termination

The app sends reminders

Post condition

The reminders stop once chore is completed.

### Requirement 8: Send Chores

#### Use case

Send Chores

#### Scope

The scope of this is for parents to send chores to their children

#### Description

This use case describes the process of sending a chore when using the app.

#### Flow Description

Precondition

The system is idle.

Activation

This use case starts when the user opens the app.

Main flow

1. The user is on the homepage.
2. The user presses the send chore button.
3. The user is then brought to a menu that allows them to choose who they want to send the chore to..
4. The user selects who they want to send the chore to.
5. The user is then shown the various chores they can send them.
6. The user selects the chore they want to send.
7. The user presses send.
8. The chore is sent via messaging app

Alternative flow

-

Exceptional flow

-

Termination

It terminates once the chores is being sent.

Post condition

The system goes to a wait state

### Requirement 9: Confirm Chore

#### Use case

Confirmation of the completion of the task

#### Scope

The scope of this use case is for the user (child) to confirm to the parent user that they have completed the chore

#### Description

This use case describes the process of confirming the completion of the chore

#### Flow Description

Precondition

The system is idle.

Activation

This use case starts when the child user completes a chore.

Main flow

1. The user will complete their chore
2. They will go into the app.
3. The will go into the ‘your chores’ menu.
4. The user then selects the chore they have completed.
5. The user then presses the ‘completed’ button.
6. The system presents a message “Chore completed”.
7. The chore will be removed from the ‘your chores’ menu

Alternative flow

-

Exceptional flow

-

Termination

An image must be sent to a parent via third party messaging app and the parent must approve.

Post condition

Child user is reminded of chore during set intervals of notifications.

### Requirement 10: Earn XP

#### Use case

Earn XP

#### Scope

The scope of this use case is for a user to earn XP

#### Description

This use case describes the XP progression for the app. As the children do more chores they earn rewards by reaching milestones.

#### Flow Description

**Precondition**

The user logs in to the system.

**Activation**

This use case starts when a user finishes a chore and is approved by the host/parent.

**Main flow**

1. User opens the app
2. User presses log in button
3. User opens the chores
4. User does the chore
5. User sends a proof-of-completion to the host
6. Host approves completion
7. User earns XP

**Alternative flow**

-

**Exceptional flow**

-

**Termination**

The system stores the XP progress

**Post condition**

The system returns the user to the main page

### Requirement 11: Synchronise Data

#### Use case

Synchronise Data

#### Scope

The scope is to Synchronise Data

#### Description

This use case describes the process of synchronising data between all devices using Firebase Real-time Databse.

#### Flow Description

Precondition

Must have online access; connected to WiFi..

Activation

This use case starts when the user opens the app.

Main flow

1. Subscribe to a certain sub/group of data
2. Data moves up and down a web-socket channel which is a permanent and full-duplex stuff.

Alternative flow

It keeps listening for any data update.

Exceptional flow

Shows message to connect to online.

Termination

No termination unless the app is close or completely disabled.

Post condition

The system goes to a wait state.

## Non-Functional Requirements

### Performance/Response time requirement

The app must be able to respond to the user’s request in a short amount of time. It must be able to send messages from one device to another instantly. The device must always stay connected to the internet and its delivery performance will depend on the internet speed of the device.

The App includes a feature that will enable parents to schedule a reminder to be sent to the child’s device. The system must be able to send the message at the time pre-set by the parents.

### Availability requirement

In the early stage of the development, the system will be more likely to have bugs, which will affect the availability of a particular function or feature. Our team will try to minimise downtime and be available to fix any problems with in college times. Our team is usually available for six hours a day during weekdays.

### Recover requirement

We use Github to maintain a version control for every progress and changes of the system. In the event of system failure during update or maintenance, we will be able to roll back the previous version of the working system. We will try minimise as much failure system as possible by unit testing before any new releases.

### Security requirement

The security of the app is very important part of the requirements. The apps sends messages from one device to another. It is very important that the message can only be received by that specific device. No unauthorised devices should be able to see the message. We are looking to implement an encryption technology on our app in the near future of the project.

### Reliability requirement

The app must be reliable in a way that users should not have a problem while using the app. The app should do what it was designed to do and everything should be as described. The objective of this requirement is to prevent or reduce the likelihood of failures within the app. The reason of the failure should be found and dealt with. The app should be ready for any kind of failures such as storage, system, software etc. The users must be happy with how the app is working.

### Maintainability requirement

It should always be maintained and up to standards. The way to maintain the app is by repairing any issues reported as soon as possible. It should work in a way that it prevents unexpected problems from happening. By maintaining the app, the security and efficiency should be maximised. The apps life should be increased as in it should be ready for all sort of software updates when needs to be.

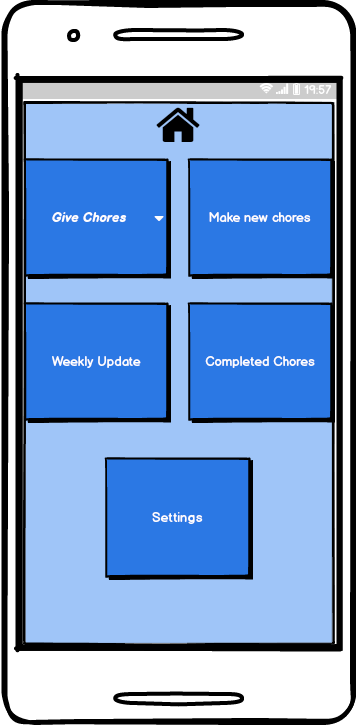
### Portability requirement

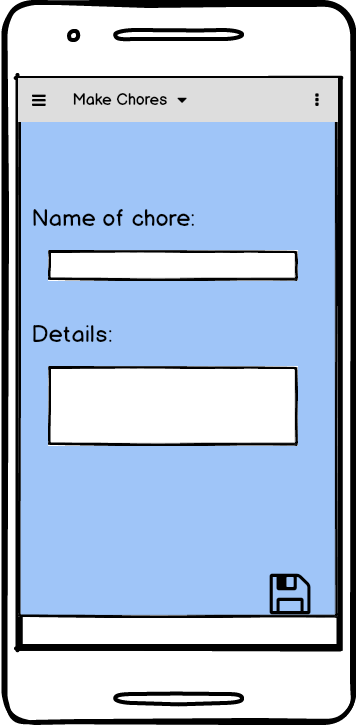
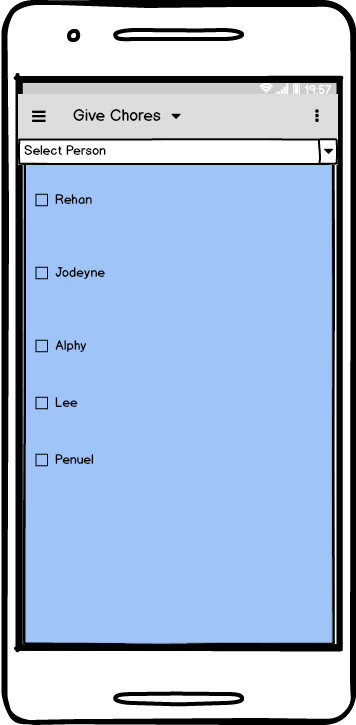
The app should be portable meaning that it should work on any android device without any problems. The goal is to make the app successful and slowly develop an iOS version of the app. The app must not fail on any device and should be compatible with all android systems from jellybean to Oreo.

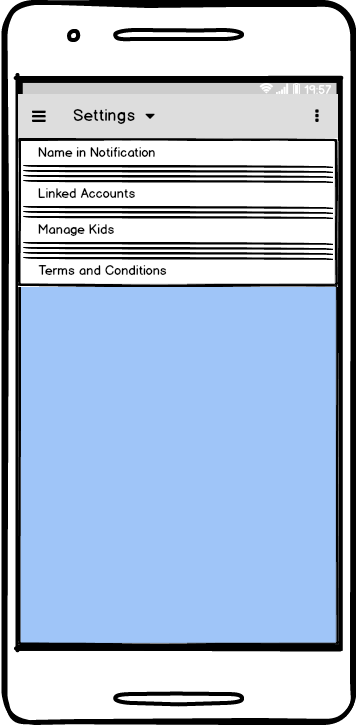
### Extendibility requirement

Our main requirement would be extendibility as we should be able launch the app on multiple platforms such as iOS and Microsoft as well as android. The way this can be done is by changing the source code of the main app. This is important as we can develop our app and launch it into multiple platforms.

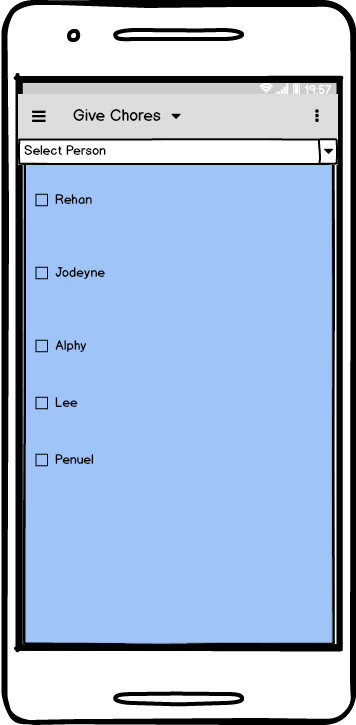
# GUI

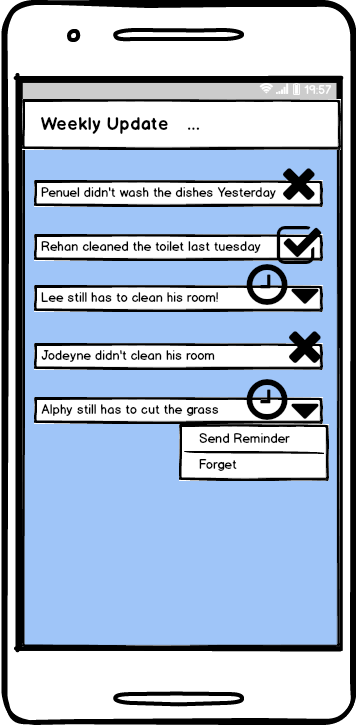












# System Architecture

|  |
| --- |
| Use a class diagram to outline the structure of the system. Explain briefly why you have chosen this architecture. You might want to use Visio or Rational Rose to create these. |

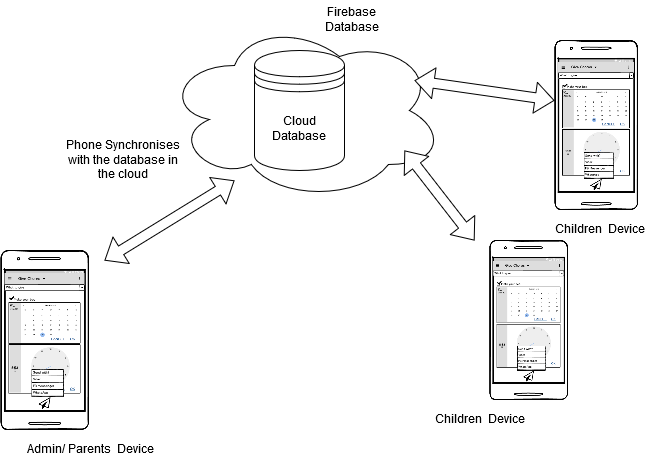
The System Architecture contains the following:

* Class Diagram
* A Conceptual Model of how Firebase Realtime Database is use for the app
* NoSQL Embedded Data Model

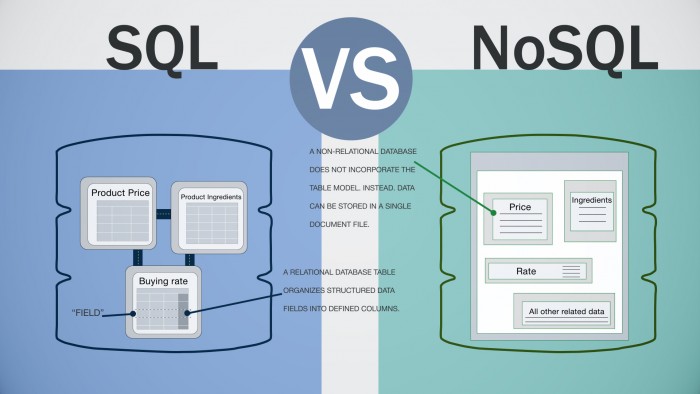
## **Database -- Data Model**

**Firebase Realtime Database**

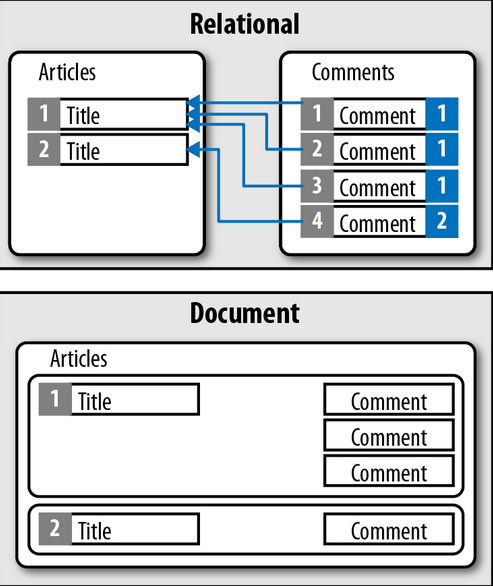
We will be using a Firebase Database to store the data. Firebase Realtime database will allow the app’s data to synchronise on all device.



**SQL vs NoSQL**

****

Source: <https://www.dbbest.com/blog/wp-content/uploads/2016/02/sql-vs-nosql-2-e1455797740330.jpg>

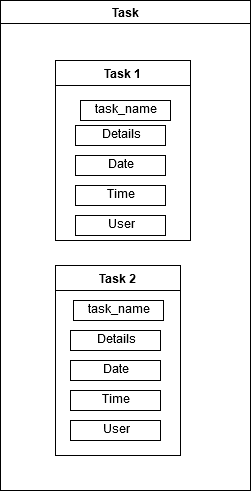


Source:

<http://prabathsl.blogspot.ie/2013/02/document-oriented-database_14.html>

**NoSQL Model Diagram**

This is the model of our NoSql. This database does not incorporate the table model, instead data are stored in a single document file.

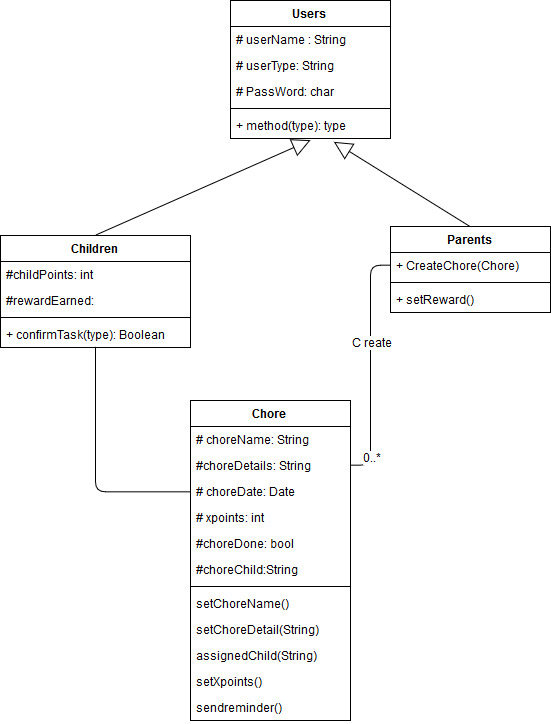


Table

**Key**

**Collection**

## **Class Diagram**

****

# System Evolution

At the moment our app only runs on android devices. The app has the potential to expand its system to run on iOS devices. The next stage would be to have a Web Application version of this App. This allows the app to be accessible on the web, desktop and any devices. An integration with other existing application is also an option; such as planner or google calendar.

In terms of purpose & usage, this could evolve into a task management system for business and project management use. The app can also be use by college students for group projects where the team leader assigns work to team members.

In 5 years time there will be new devices and technologies. We will have to keep up with the trend and make sure that our app will have new features and will run on all sort of devices. The application could be implemented on home appliances devices such as washing machines, fridge etc. In a futuristic outlook, we could also implement our application to a robot; where you can assign chores to the robot; to do the daily routine.

# References & Bibliography

* Project Scope
  + freeCodeCamp. 2018. How to effectively scope your software projects – freeCodeCamp. [ONLINE] Available at: <https://medium.freecodecamp.org/how-to-effectively-scope-your-software-projects-from-planning-to-execution-e96cbcac54b9>. [Accessed 05 February 2018].
* Software Testing
  + CodeProject. 2018. What is software testing? What are the different types of testing? - CodeProject. [ONLINE] Available at: <https://www.codeproject.com/Tips/351122/What-is-software-testing-What-are-the-different-ty>. [Accessed 05 February 2018].
* Definition, Acronyms & Abbreviation
  + freeCodeCamp. 2018. What is an API? In English, please. – freeCodeCamp. [ONLINE] Available at: <https://medium.freecodecamp.org/what-is-an-api-in-english-please-b880a3214a82>. [Accessed 05 February 2018].
  + SearchSoftwareQuality. 2018. What is integrated development environment (IDE)? - Definition from WhatIs.com. [ONLINE] Available at: <http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment>. [Accessed 05 February 2018].
* Database - SQL vs NoSQL
  + Images
    - DB Best Chronicles. 2018. Database decisions for the Internet of Things. [ONLINE] Available at: <https://www.dbbest.com/blog/database-decisions/>. [Accessed 09 February 2018].
    - Document Oriented Database ~ Prabath's Tech Blog. 2018. Document Oriented Database ~ Prabath's Tech Blog. [ONLINE] Available at: <http://prabathsl.blogspot.ie/2013/02/document-oriented-database_14.html>. [Accessed 09 February 2018].
* UML Class Diagram
  + tutorialspoint.com. 2018. Java Date and Time. [ONLINE] Available at: <https://www.tutorialspoint.com/java/java_date_time.htm>. [Accessed 12 February 2018].
* What is User Requirement
  + Coley Consulting - Principal Consultant: Phil Coley. 2018. User Requirements - three areas to include and one to avoid.. [ONLINE] Available at: <http://www.coleyconsulting.co.uk/require.htm>. [Accessed 10 February 2018].