School Parent App

Contents

[1.0 Problem statement 2](#_Toc65488937)

[2.0 Architecture Diagram for the Problem Statement 2](#_Toc65488938)

[3.0 Use case details 3](#_Toc65488939)

[4.0 Functional/Non-Functional Requirement of the Problem Statement 4](#_Toc65488940)

[5.0 Milestone 7](#_Toc65488941)

[6.0 Skills to develop the project 8](#_Toc65488942)

[7.0 Implementation Notes 8](#_Toc65488943)

[8.0 Evaluation rubrics 9](#_Toc65488944)

# Problem statement

The purpose of the requirements document is to systematically capture requirements for the project and the system “School Parent App” to be developed. The application should be Cloud Native Architecture with Microservices. Both functional and non-functional requirements are captured in this document. It also serves as the input for the project scoping.

**About the System**

The client would like to develop an application called School Parent App to capture the details of the students who likes to get the online updates such as Fee details, circulars, and holiday list.

**Scope of the System**

The scope of the system is explained through its modules as follows

* Register Student and Parent Information - used by the parents/students to enter the details into the application. The system stores the details of the student in the system and able to edit it.
* Process the application by Staff – The School Staff should be able to review the details submitted by the parent and approve/reject it to proceed to the further process.

# Architecture Diagram for the Problem Statement

**Use case Diagram**

**US\_01 Parent Information**

Submitted the Information

Parent

Save to DB

**US\_02 Update parent/student details**

Update the details

Parent

Save to DB

**US\_03 Staff Review**

School Staff

Review the Parent Information and processing the application

Save to DB

**US\_04 Create Circular**

School Staff

Create the circular about the important updates

Save to DB

**US\_05 View Circular**

Parent

View the circular about the important updates and acknowledge

Save to DB

# Use case details

|  |  |  |
| --- | --- | --- |
| **User Story #** | **User Story Name** | **User Story** |
| US\_01 | Parent Information | As a parent user, I should be able to enter my details in the system.  Acceptance criteria:  Parent user should be able enter the details in the system and it should be saved in the database.  Capture the details like Parent and Student Name, Student Register Number, Address, State, Country, City, Zip code, Email Address, Primary Contact Person, Primary Contact Person Mobile, Secondary Contact Person, Secondary Contact Person Mobile, |
| US\_02 | Update the Parent Information | As a Parent User, I should be able to update the personal details before submitting. |
| US\_03 | School Staff Review the Application | As a School Staff, I should be able to review the student information and Approve/Reject the application |
| US\_04 | Create Circular | As a School Staff, I should be able to create the circular about the important updates |
| US\_05 | View Circular | As a Parent User, I should be able to view the circular details and acknowledge. |

# 

# Functional/Non-Functional Requirement of the Problem Statement

US -01 Parent Information

Business Rules & Validations

1. Registration id should be generated automatically during the time of submitting the application and should be shown in the success message.
2. The Student and Parent name should contain only alphabets and space.
3. Student Register Number id should be in the format of ‘R-XXX’.XXX should be match with ID card given by School.
4. All fields are mandatory.
5. Zip code should have only numbers and 6 digits
6. City should contain only alphabets
7. Primary Contact Person Mobile, Secondary Contact Person Mobile should be 10 digits.
8. Email id should contain @ and. symbols.
9. The status of the application should be saved as “Submitted”

**Validations**

* All fields are mandatory.
* Based on the country, state must be populated in the dropdown automatically.
* Age should be greater than 4.
* Registration Date should not be lesser than system date.

Update the Parent Information

|  |  |
| --- | --- |
| US\_02 | Update the Parent Information |
| Description  The Parent users should be able to update the details before submitting. | |
| Input Parameters  Below are the input parameters.  Parent Name, Student Register Number, Address, State, Country, City, Zip code, Email Address, Primary Contact Person, Primary Contact Person Mobile, Secondary Contact Person, Secondary Contact Person Mobile, | |
| Business Rules & Validations   * All fields are mandatory. * Registration id should be generated automatically during the time of submitting the application and should be shown in the success message. * The Student and Parent name should contain only alphabets and space. * Student Register Number id should be in the format of ‘R-XXX’.XXX should be match with ID card given by School. * All fields are mandatory. * Zip code should have only numbers and 6 digits * City should contain only alphabets * Primary Contact Person Mobile, Secondary Contact Person Mobile should be 10 digits. * Email id should contain @ and. symbols. * The status of the application should be saved as “Submitted” | |

Processing the Application

|  |  |
| --- | --- |
| US\_03 | School Staff Processing the Application |
| Description  Staff should be able to see the List of Parents information and review their details and approve or reject the application. | |
| Input Parameter | |
| Business Rules & Validations   * Staff should not be able to edit the Parent information, they can only view the details * Staff should be able to approve or reject the application and accordingly the status should be updated. | |

Create the circular

|  |  |
| --- | --- |
| US\_04 | School Staff Create the circular |
| Description  Staff should be able to create the circular about the important updates from school | |
| Input Parameter  Below are the input parameters.  Notification Date, Information text, Notification Posted By | |
| Business Rules & Validations   * All fields are mandatory. * NotificationDate should not be lesser than system date | |

|  |  |
| --- | --- |
| US\_05 | View the circular |
| Description  Parent should be able to see the List of circular information and review the details and acknowledge it. | |
| Input Parameter | |
| Business Rules & Validations   * Acknowledge Date should not be lesser than system date | |

**Service Requirements**

**US\_01 Parent Information:**

Once the Parent user enters the details, they should be sent to the POST method and saved in the db.

Mandatory fields should be validated as mentioned in the rules above and 400 exception response should be sent with the missing field details.

When the details are saved successfully, the service should response 200 ok along with success message.

Registration ID should be generated automatically when the user submits the application and should be a numeric of 10 digits

If there are any exceptions while connecting/saving to DB, the service should throw corresponding error with error status as 500.

**US\_02 Update the Parent Information:**

To retrieve the details of the Parent, a GET method should be implemented to fetch the details.

Once the user enters the details, they should be sent to the PUT method and saved in the db.

Mandatory fields should be validated as mentioned in the rules above and 400 exception response should be sent with the missing field details.

When the details are saved successfully, the service should response 200 ok along with success message.

If there are any exceptions while connecting/saving to DB, the service should throw corresponding error with error status as 500.

**US\_03 Staff Processing the Application:**

To retrieve the list of Parent and the details of the Parent, a GET method should be implemented to fetch the details.

Once the Staff validates the details and approve/reject, they should be sent to the PUT method and saved in the db.

Mandatory fields should be validated as mentioned in the rules above and 400 exception response should be sent with the missing field details. When the details are saved successfully, the service should response 200 ok along with success message.

If there are, any exceptions while connecting/saving to DB. The service should throw corresponding error with error status as 500.

**US\_04 School Staff Create the circular:**

Once the Staff user enters the circular details, they should be sent to the POST method and saved in the db.

Mandatory fields should be validated as mentioned in the rules above and 400 exception response should be sent with the missing field details.

When the details are saved successfully, the service should response 200 ok along with success message.

Registration ID should be generated automatically when the user submits the application and should be a numeric of 10 digits

If there are any exceptions while connecting/saving to DB, the service should throw corresponding error with error status as 500.

**US\_02 View the circular:**

To retrieve the details of the circular, a GET method should be implemented to fetch the details.

Once the user acknowledges the details, they should be sent to the PUT method and saved in the db.

Mandatory fields should be validated as mentioned in the rules above and 400 exception response should be sent with the missing field details.

When the details are saved successfully, the service should response 200 ok along with success message.

If there are any exceptions while connecting/saving to DB, the service should throw corresponding error with error status as 500

**Expected Deliverables**

The following deliverables are expected as outcomes:

* Application Code base
* Readme document on the complete application
  + Setup of the application
  + How to run the application
  + Any inference
  + Screenshot of UI results
* Reports:
  + Unit/Functional Test Report

# Milestone

The milestone for the project use is given below

|  |  |  |
| --- | --- | --- |
| Milestone | Duration (in weeks) | Topic |
| Milestone -1 |  | Develop the required APIs for the application  C:\Users\366598\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\C9B327B2.tmp  Or Develop all the APIs with NodeJS. |
| Milestone - 2 |  | Design and develop the UI for the application |
| Milestone -3 |  | Integrate service layer with UI component |

# Skills to develop the project

List the Technology based on your respective technology stack, that will be used to development the project.

Associate will choose any one of the technology stack and develop the application.

|  |  |
| --- | --- |
| Skill Stack | Java |
| Front end | Angular /React  Bootstrap/ CSS  JavaScript/ JQuery  Typescript  Karma/ Cypress/ Jest |
| Service End | Spring Boot, Spring MVC, JDK, Maven/ SonarQube/ Junit or NodeJS/Express JS |

# Implementation Notes and Rubrics

As per the project requirement modification can be done in the below table.

|  |  |
| --- | --- |
| Milestone -1 | **SpringBoot:**   * Create Spring Boot REST Microservice to perform SAVE Operation using POST method. * Use Microservice Architecture * Follow coding standards * Follow Standard project structure * Message input/output format should be in JSON (Read the values from the property/input files, wherever applicable). Input/output format can be designed as per the discretion of the participant * Database connections and web service URLs should be configurable. * Use browser / POST Man to invoke APIs * Run SonarQube for code quality. * Implement Junit for unit testing. * Use Mockito framework wherever appropriate. |
| Milestone-2 | * Implement user-stories using any one of the UI frameworks [Angular/React] * Design application with Minimum Backend or Mock backend as the main focus in on frontend skills * Implement Forms, databinding, validations   Use Appropriate unit test framework. |
| Milestone -3 | * Integrate service layer with UI component. |

# 8.0 Evaluation rubrics

|  |  |
| --- | --- |
| Angular | * Build clean and robust application * Dependency Injection * SOLID Principles and Design Patterns * Naming Conventions and standards * Exception Handling * Logging * Performance Considerations * Use Angular CLI * Break down into small reusable components * Maintain proper folder structure * Follow consistent Angular coding styles Utilize ES6 Features * Use Lazy Loading * Implement the lifecycle hook interface * Cache API calls * State Management * Always Document * Aliases for imports * Using Sass * Save Memory Bytes by using AsyncPipes |
| REACT | * Build clean and robust application * Best practice to build clean architecture application * Maintainability of the application * Dependency Injection * SOLID Principles and Design Patterns * Documentation * Naming Conventions and standards * Exception Handling * Logging * Performance Considerations * Keep components small and function-specific,Functional Components over Class Components * Use Higher Order Components * Use PropTypes * Reusability,keep creation of new components to the minimum required and Consolidate duplicate code * Follow Naming conventions * Separate stateful aspects from rendering * Follow Standard Project structure * Use snippet libraries -ES7 React, Redux, JS Snippets, etc. Write tests for all code |
| Typescript/JavaScript | * Apply the best practices * Avoid common pitfalls and mistakes other JavaScript developers make * Write solid JavaScript code |
| Java | * Lambda Expressions * Functional Interfaces * Default methods in Interface * Static Methods in Interfaces. * Predicate * Function * Consumer * Supplier * Method Reference & Constructor Reference by Double Colon(::) Operator * Stream API * Date & Time API ( Joda API) * **Demonstrate 2 Debugging and Troubleshooting**  1. Debug a Memory Leak 2. Understand the OutOfMemoryError Exception 3. Troubleshoot a Crash Instead of OutOfMemoryError 4. Diagnose Leaks in Java Language Code 5. Diagnose Leaks in Native Code |
| Spring Boot | * Perform CRUD operations against a in memory db using Spring Data JPA * Perform CRUD operations against Database * Expose out REST APIs using Spring Web * Spring boot auto configures a spring project * Enable health metrics for the application * Customize health metrics endpoint with your own information Use Spring Boot Profiles * **Demonstrate 2 Debugging and Troubleshooting** (Create two scenarios of troubleshooting your Spring Boot application in your project demo) |
| Microservices | * Separate Data Store for Each Microservices * Keep all code in a Microservices at a similar level of maturity and stability * Do a Separate Build for Each Microservices * Deploy in Containers * Servers as Stateless * Use Fault Tolerance techniques * Use the API gateway * Use Domain-Driven Design * Focus on data and API security * Use Distributed Configuration * Monitor microservices, Use Monitoring Tools * App metrics and health checks * Send Logs to a Centralized Location * Structure Log Data * Add Context to Every Request * Include a Unique ID in the Response * Use OAuth for user identity and access control * Use a distributed firewall with centralized control |
| NodeJS/ExpressJS | * NodeJS, Helmet and Express JS should be used in the * API’s application * Advantage of using NodeJS, Projections in Mongo Queries over other frameworks. |