­­MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Cloud-based Backend as a Service for Building Mobile Applications

|  |  |
| --- | --- |
| Nhóm 7 | |
| Group member | Nguyễn Mạnh Hùng – SE61388 Nguyễn Hữu Lộc – SE61312  Phạm Bảo Toàn – SE61011  Vũ Văn Quyết – SE61071 |
| Supervisor | Kiều Trọng Khánh |
| Ext. Supervisor | N/A |
| Capstone Project code | CSBM |

-Ho Chi Minh, *11-05-2016*-

**Table of Contents**

[**A. Introduction** 5](#_Toc451967954)

[**1.** **Project Information** 5](#_Toc451967955)

[**2.** **Introduction** 5](#_Toc451967956)

[**3.** **Current Situation** 5](#_Toc451967957)

[**4.** **Problem Definition** 5](#_Toc451967958)

[**5.** **Proposed Solution** 6](#_Toc451967959)

[**5.1. Feature functions** 6](#_Toc451967960)

[**5.2. Advantages and Disadvantages** 6](#_Toc451967961)

[**6.** **Functional Requirements** 6](#_Toc451967962)

[**7.** **Role and Responsibility** 7](#_Toc451967963)

[**B. Software Project Management Plan** 8](#_Toc451967964)

[**1.** **Problem Definition** 8](#_Toc451967965)

[**1.1. Name of this Capstone Project** 8](#_Toc451967966)

[**1.2. Problem Abstract** 8](#_Toc451967967)

[**1.3. Project Overview** 8](#_Toc451967968)

[**2.** **Project organization** 11](#_Toc451967969)

[**2.1. Software Process Model** 11](#_Toc451967970)

[**2.2. Roles and responsibilities** 12](#_Toc451967971)

[**2.3. Tools and Techniques** 13](#_Toc451967972)

[**3.** **Project Management Plan** 13](#_Toc451967973)

[**3.1. Product Backlog** 13](#_Toc451967974)

[**3.2. Deliverables** 14](#_Toc451967975)

[**3.3. All Meeting Minutes** 15](#_Toc451967976)

[**4.** **Coding Convention** 15](#_Toc451967977)

[**C. Software Project Management Plan** 16](#_Toc451967978)

[**1.** **User Requirement Specification** 16](#_Toc451967979)

[**1.1. Guest Requirement** 16](#_Toc451967980)

[**1.2. User Requirement** 16](#_Toc451967981)

[**1.3. Authenticated user Requirement** 16](#_Toc451967982)

[**2.** **System Requirement Specification** 16](#_Toc451967983)

[**2.1. External Interface Requirement** 16](#_Toc451967984)

[**2.2.** **System Overview Use Case** 17](#_Toc451967985)

[**2.3. List of Use Case** 18](#_Toc451967986)

[**3.** **Software System Attribute** 27](#_Toc451967987)

[**3.1. Usability** 27](#_Toc451967988)

[**3.2. Reliability** 27](#_Toc451967989)

[**3.3. Availability** 27](#_Toc451967990)

[**3.4. Security** 27](#_Toc451967991)

[**3.5. Maintainability** 27](#_Toc451967992)

[**3.6. Portability** 28](#_Toc451967993)

[**3.7. Performance** 28](#_Toc451967994)

[**4.** **Conceptual Diagram** 28](#_Toc451967995)

**Definitions, Acronyms, and Abbreviations**

|  |  |
| --- | --- |
| Name | Definition |
| BaaS | Back-end as a service |

# **A. Introduction**

## **Project Information**

- Project name: **Cloud-based Backend as a Service for Building Mobile**

**Applications.**

- Project code: **CSBM.**

- Project type: **Web Application and Mobile Framework.**

- Start date: **11-05-2016.**

- End date: **21-8-2016.**

## **Introduction**

In this document, we introduce a solution for mobile developers. Developing mobile applications is more and more popular and cloud computing also extremely develops. The combination is very complex to build the effective application on smartphone. Based on our researches and analysis, we proposed a cloud-based backend as a service for mobile developers.

We provide a service that includes website application and mobile framework, which helps mobile developers to develop their application quickly, easily to change, and flexibility in modification. Somethings supported the developer building their application without worrying about the backend services.

## **Current Situation**

When developing mobile applications, developers need to concern too much about back-end such as: creating service support push notification, creating and managing database mobile… Most modern mobile applications store data and interact with other services on the internet. User accounts, shared content, documents and purchases; these things all need to be stored somewhere else, and cloud is a good solution. There is a cloud-based backend as a service of Facebook named “Parse” (https://www.parse.com), however the “Parse” is shutdown in first month 2016…

## **Problem Definition**

Others backend as a service:

* Advantages:
* Parse, Firebase… are well-known and already had much users.
* Disadvantages:
  + Parse (http://parse.com/) is shutdown in first month 2017.
  + CloudKit (<https://developer.apple.com/icloud/>) is limited to iOS and require user to use iCloud.
  + Firebase (<https://www.firebase.com/>) hasn’t push notification

Common mobile development:

* Advantages:
* Taking control everything from mobile layout to back-end component.
* Disadvantages:
* Mobile developers need to have the knowledge about web service or wait provided services from web service developer.
* Mobile developers hard to focus on creating user experience.
* Mobile application development is waste of time on building API.

## **Proposed Solution**

Our proposed solution is to develop the Cloud-based Backend as a Service named “CSBM” to support mobile developer manage application data via our provided framework and only focus on creating extraordinary user experiences. We’ll take care of the rest.

### **5.1. Feature functions**

* Web application: design for developer to manage data and configuration of their mobile application.
* Server: parse data from client side to store and sync data with our NoSQL cloud database.
* Mobile framework: provide framework for mobile developer to interact with our database.

### **5.2. Advantages and Disadvantages**

* **Advantages:**
  + Real-time database: data is stored in cloud database and synced to all connected clients in real-time.
  + Push notification: auto push notification to all connected client after data is changed.
  + Mobile framework for developers to interact with their data on server database.
* **Disadvantages:**
  + Security: users only manage their data while the security is based on the service provided.
  + Application downtime: users can’t control downtime when CSBM server has problem.

## **Functional Requirements**

* **Server component**
  + Parse data from client dashboard to system storage.
  + Notify to connected client.
  + Sync database to server automatically.
* **Web component**
  + Manage applications.
  + Manage data.
* **Mobile framework** 
  + Provide framework for mobile.

## **Role and Responsibility**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Full Name | Role | Position | Contact |
| 1 | Kiều Trọng Khánh | Project Manager | Supervisor | khanhkt@fpt.edu.vn |
| 2 | Nguyễn Mạnh Hùng | Developer | Leader | hungnmse61388@fpt.edu.vn |
| 3 | Nguyễn Hữu Lộc | Developer | Member | locnhse61312@fpt.edu.vn |
| 4 | Vũ Văn Quyết | Developer | Member | quyetvv61071@fpt.edu.vn |
| 5 | Phạm Bảo Toàn | Developer | Member | toanpbse61011@fpt.edu.vn |

*Table 1: Roles and Responsibilities*

# **B. Software Project Management Plan**

## **Problem Definition**

### **1.1. Name of this Capstone Project**

- Official name: Cloud-based Backend as a Service for Building Mobile Applications.

- Vietnamese name: Dịch vụ đám mây hỗ trợ developer xây dựng mobile application.

- Abbreviation: CSBM.

### **1.2. Problem Abstract**

At the moment, especially in Viet Nam, not much developer use BaaS (Backend as a Service). The development of mobile application include building back-end so it requires mobile developer have knowledge about coding web api or they may wait for web api from others developers. This situation makes mobile developers can’t focus on creating good user experiences, and then the effective of build a mobile application is always not best liked developer hope.

Besides, currently there are Parse (<http://parse.com/>), Cloud-Kit (<https://developer.apple.com/icloud/>), Firebase (<https://www.firebase.com/)>,... But they have not provided developers a good way to build a mobile application. Cloud-Kit is limited to iOS and require user to use iCloud. Firebase hasn’t push notification. Parse is stop service provider in first month 2016 and is shutdown in first month 2017, but many applications are there. If Parse shutdown, developers who are using Parse will have difficulty in finding another BaaS like Parse.

For the goal that improving mobile development, we provide a service which base on Parse Server, to reduce the time consuming in mobile development and mobile developers can focus more on user interface and user experiences. By using Parse Server, we develop a website to create new applications, and a service to sharing data between user’s different devices. The service also haves push notifications, if either of those are important to developer applications, our service will likely save develop time.

### **1.3. Project Overview**

#### **1.3.1. Current Situation**

Below are the problems encountered in this project:

* + **Disadvantages:**
    - Parse source code was developed with NodeJs that nobody in team has much knowledge about.
    - Parse source code not public function to allow users create new application.
    - Parse server now can only run one application. If developers have many application they will need many Parse server to run all of them.
    - Team hasn’t much knowledge about how to code and build a mobile framework.
  + **Advantages:**
    - The server is ready to run based on Parse source code.

#### **1.3.2. The Proposed System**

According to technology researches, we found out that Parse server of Facebook can help us resolving the problem of reducing time consuming in mobile development and also help developers who are using Parse now can move to and continue develop application.

To resolve the problem that Parse uses NodeJs as main language for the server, we can use NodeJs Tutorial from <http://www.tutorialspoint.com/nodejs/> for understanding how NodeJs work. We also find the necessary help from NodeJs development forum to solve problems.

We must to know what dashboard send request to server to create a new application to write a function allow developer can create new application from our service dashboard. According to technology researches, we decided AngularJs will be the framework for dashboard.

We assign responsibility in horizontal, it means everyone will do a whole big think and separated from the others. This decision based on that everything about this project is really new for all of the member so let each one focus on one problem is the best way.

##### **1.3.2.1. Website (Dashboard)**

Main web application is a place for developer to create a new application or register account to use main function of the service website. Developer also can use this website to management the data of their application.

* For developer
  + - Create a new application: Developer can create a new application.
    - Manage data: Create/Edit/Remove developer data in application.
  + For guest
    - Register: User can register new account to use the service.

##### **1.3.2.2. Mobile Framework**

This framework is use for Developer’s mobile Application, which provide APIs to control mobile application data. Mobile developers can control their data really easily after download framework which appropriate to their language (Android or iOS) and apply to their application.

#### **1.3.3. Boundaries of the System**

* This service is built basing on Parse server of Facebook. Our main target is improving the mobile development and help developers, whose applications is using Parse now, have an appropriate place to migrate to.
* The complete product includes:
  + A web application that allow:
    - Register account to use main function of the system
    - Create a new application
    - Manage data
  + Mobile Framework that allow:
    - Provide API

#### **1.3.4. Future Plans**

Current service only support for mobile. In future we will:

* + - Provide framework for website developers.

#### **1.3.5. Development Environment**

##### **1.3.5.1. Hardware requirements**

* + For server

|  |  |  |
| --- | --- | --- |
| Hardware | Minimum Requirements | Recommended |
| Internet Connection | 512Kbps | 4 Mbps |
| Operating System | Ubuntu Server 12 LTS | Ubuntu Server 14.04.2 LTS |
| Computer Processor | Intel® CORE i3 Quad core 2.1 GHz | Intel® CORE i7 Quad core 2.4 GHz |
| Computer Memory | 1GB of RAM | 4GB of RAM or more |

*Table 2: Hardware Requirement for server*

* + For website development

|  |  |  |
| --- | --- | --- |
| Hardware | Minimum Requirements | Recommended |
| Internet Connection | 512Kbps | 4 Mbps |
| Operating System | Ubuntu Server 12 LTS | Ubuntu Server 14.04.2 LTS |
| Computer Processor | Intel® CORE i3 Quad core 2.1 GHz | Intel® CORE i7 Quad core 2.4 GHz |
| Computer Memory | 1GB of RAM | 4GB of RAM or more |

*Table 3: Hardware Requirement for website development*

* + For mobile framework development

|  |  |  |
| --- | --- | --- |
| Hardware | Minimum Requirements | Recommended |
| Internet Connection | 512Kbps | 4 Mbps |
| Operating System | Ubuntu Server 12 LTS | Ubuntu Server 14.04.2 LTS |
| Computer Processor | Intel® CORE i3 Quad core 2.1 GHz | Intel® CORE i7 Quad core 2.4 GHz |
| Computer Memory | 1GB of RAM | 4GB of RAM or more |

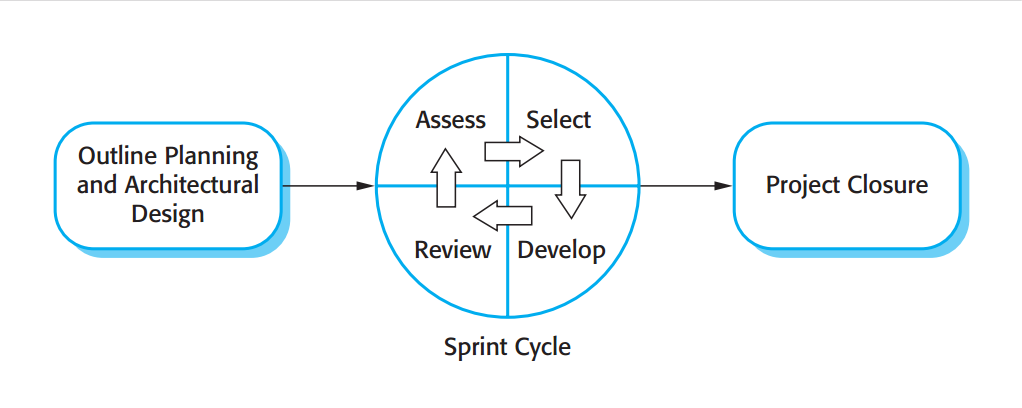
*Table 4: Hardware Requirement for mobile framework development*

## **Project organization**

### **2.1. Software Process Model**

The project is developed under Scrum model. We choose this model because the following reasons:

* + This project is about Parse server, NodeJs and building mobile framework these are really new to all of member. That why our team need focus on research, study and try our best to implement a core flow of this project early.
  + We need to provide many demo for each requirement to make sure that our team can complete this project.
  + This project is about building framework so it’s really hard to finish the document first. We need coding through some demo to understand more about the operation mechanism of Parse server and figure out how to build framework effectively.



*Figure 1: Scrum model (Source: Software Engineering, 9th Edition, Chapter 3, Figure 3.8)*

### **2.2. Roles and responsibilities**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in Group** | **Responsibilities** |
| 1 | Mr. Kiều Trọng Khánh | Scrum Product Owner – Technical Expert | * Specify user requirement * Specifying the business * Control the development process * Give advices on techniques, solutions and business analysis support |
| 2 | Nguyễn Mạnh Hùng | Scrum Master, Business Analyst, Developer, Tester | * Managing process * Researching solutions and techniques * Assigning task for members * Study Parse server for building “Create new app”. |
| 3 | Nguyễn Hữu Lộc | Team Member, Business Analyst, Developer, Tester | * Researching solutions and techniques * Build framework for iOS developers. |
| 4 | Vũ Văn Quyết | Team Member, Business Analyst, Developer, Tester | * Researching solutions and techniques * Build framework for Android developers. |
| 5 | Phạm Bảo Toàn | Team Member, Business Analyst, Developer, Tester | * Researching solutions and techniques * Build Dashboard |

*Table 5: Roles and Responsibilities Details*

### **2.3. Tools and Techniques**

|  |  |
| --- | --- |
| **Tools** | **Name / version** |
| Html, Javascript Editor | Sublime Text V3 |
| iOS IDE | Xcode 7.2 |
| Android IDE | Android Studio 2.1 |
| DBMS | Robomongo 0.9.0 RC8 |
| Source control | GitHub |
| Document | Microsoft Word 2016 |

*Table 6: Tools used for this project*

|  |  |
| --- | --- |
| **Techniques** | **Name / version** |
| Backend | NodeJs 6.2.0 |
| Frontend | AngularJs 1.5.0 |
| iOS Framework | Objective-C 2.0 |
| Android Framework | Java |
| DBMS | MongoDB 3.2.6 |

*Table 7: Techniques used for this project*

## **Project Management Plan**

### **3.1. Product Backlog**

|  |  |  |
| --- | --- | --- |
| **Id** | **User Story** | **Priority** |
| 1 | Create new application | High |
| 2 | Manage applications | High |
| 3 | Manage user account for an application | High |
| 4 | Mange user role for an application | Medium |
| 5 | Multiple application on one Parse server | High |
| 6 | Push notification | Low |
| 7 | Framework for Android | High |
| 8 | Framework for iOS | High |
| 9 | Application search | Low |
| 10 | Login | High |
| 11 | Logout | Low |
| 12 | Auto generate ApplicationId | Medium |
| 13 | Analytics | Low |

### **3.2. Deliverables**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Deliverable** | **Delivery date** | **Delivery Location** | **Note** |
| 1 | Introduction Document | 12/5/2016 | FPT-CMS | Report No.1 |
| 2 | Software Project Management Plan | 16/5/2016 | FPT-CMS | Report No.2 |
| 3 | Software Requirement Specification | 30/5/2016 | FPT-CMS | Report No.3 |
| 4 | Application Demo 1 | 16/6/2016 | FPT-University | Core flow demo |
| 5 | Application Demo 2 | 21/7/2016 | FPT-University | Features demo |
| 6 | Design 1 | 21/7/2016 | FPT-CMS | Report No.4 |
| 7 | Application Demo 3 | 7/8/2016 | FPT-University | Semi-final application demo |
| 8 | Design 2 | 7/8/2016 | FPT-CMS | Report No.5 |
| 9 | Completed Document | 14/8/2016 | FPT-CMS | Report No.6 |

### **3.3. All Meeting Minutes**

All meeting minutes here: <https://github.com/hungnm2904/CSBM/tree/report/Meeting%20Minutes>.

## **Coding Convention**

Summary:

* Naming Convention:
  + Variable names should be short yet meaningful. The choice of a variable name should be designed to indicate to the casual observer the intent of its use.
  + Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.
  + All names start with a letter.
  + Variable and function names written as **camelCase**.
* Declarations Convention:
  + One declaration per line is recommended since it encourages commenting.

Using Java Code Convention from:

<http://www.oracle.com/technetwork/java/codeconvtoc-136057.html>

<http://www.w3schools.com/js/js_conventions.asp>

<https://github.com/mgechev/angularjs-style-guide>

Programming with Objective-C summary:

* Naming Convention:
  + Class names must be unique across an entire app.
  + Method names should be expressive and unique within a Class.
  + Local Variables must be unique within The same scope.
  + Accessor method names Must follow conventions.
  + Object creation method names must follow conventions.

Using Object Convention from:

<https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Conventions/Conventions.html#//apple_ref/doc/uid/TP40011210-CH10-SW1>

# **C. Software Requirement Specification**

## **User Requirement Specification**

### **1.1. Guest Requirement**

Guest is a person who does not have access the system. Guest can use some functions in the system. To use all functions, guest must login. These are some functions guest can use:

* Login
* Register

### **1.2. User Requirement**

User is guest who already login to the system by his/her account and uses service of system. The customer can use some following functions:

* Manage application: Create New Application, Delete Application, Active Application.
* Manage class: Create Class, Update Class, Delete Class.
* Query data

### **1.3. Admin Requirement**

Person who manages account. Administrator can use some following functions:

* Change User status.
* Change Application status.

### **1.4. Authorized User Requirement**

Authorized User is the person who has accessed the system, besides the functions that user can use base on their role, authenticated user also can use the following function:

* Logout.

## **System Requirement Specification**

### **2.1. External Interface Requirement**

#### **2.1.1. User interface**

* The user interface use English language in web application and mobile framework.
* The user interface for website display best on 1024x768-screen size.

#### **2.1.2. Hardware interface**

* Computer:
  + OS: Ubuntu 14 LTS
  + CPU: Intel® CORE i3 Quad core 2.1 GHz
  + RAM: 1GB
* Mobile:

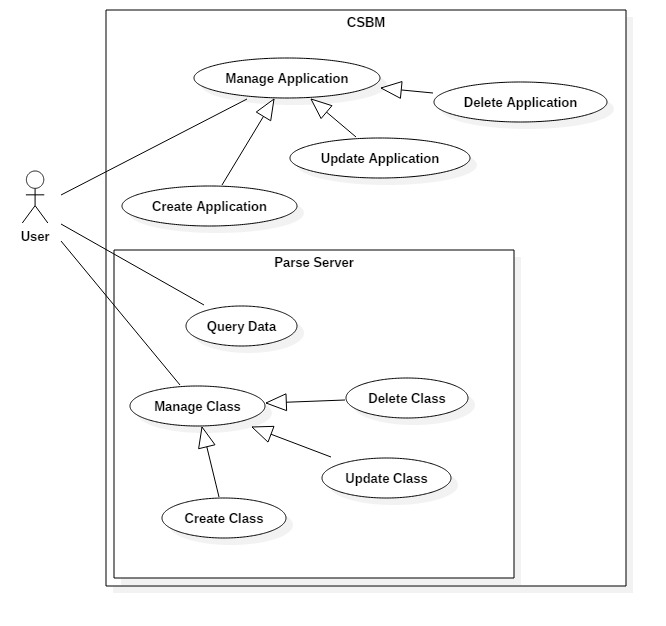
#### **2.1.3. Software Interface**

* Web application: work with Chrome (v47 or above), Internet Explorer (v10 or above), Firefox (v43 or above)
* Mobile framework: Android operation system (v 4.4 or above).

#### **2.1.4. Communication Protocol**

* Use Rest API for communication between the web browser and server.
* Use Rest API for communication between the mobile framework and server.

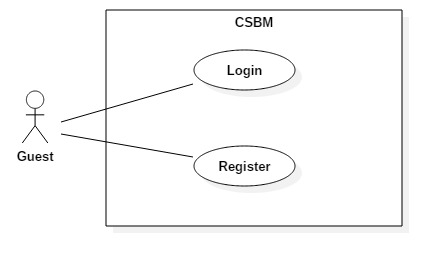
### **System Overview Use Case**



*Figure: System Overview Use Case*

### **2.3. List of Use Case**

#### **2.3.1. <Guest> Overview use case**



*Figure: <Guest> Overview Use Case*

##### **2.3.1.1. <Guest> Register**



*Figure: <Guest> Register*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC001** | | | |
| **Use Case No.** | 001 | **Use Case Version** | 1.0 |
| **Use Case Name** | Register | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * Guest   **Summary:**   * This use case allows guest register an account.   **Goal:**   * Create new account.   **Triggers:**   * Guest sends command to register.   **Preconditions:** N/A.  **Post Conditions:**   * **Success:** Guest will be logged in system with their roles. * **Fail:** Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest sends command to request register | System requires information from Guest:   * Email: free text input, required, regex [^[\_A-Za-z0-9-\\+]+(\\.[\_A-Za-z0-9-]+)\*@"+"[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$] * Full name: free text input, required, length (10-50) * Username: free text input, required, length (9-20) * Password: free text input, required, length (6-12) * Repeat password: free text input, required, length (6-12) * Phone: free number input, required, length (10-12) positive integer, value: [0,9] | | 2 | Guest inputs information |  | | 3 | Guest sends command to register  [Alternative 1]  [Alternative 2] | System shows login view  Account registered  [Exception 1]  [Exception 2]  [Exception 3]  [Exception 4] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest sends command to reset | System reset all field to blank | | 2 | Guest send command to back to login view | System shows login view  Account isn’t created |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Guest input already exist username | System show warning message “User name already exist” | | 2 | Guest input already exist email | System shows warning message “Email already exist”. | | 3 | Guest does not input required field. | System notices that guest need to input all these field:   * “Email”: System display warning message: “please fill out this field”. * “Full name”: System display warning message: “please fill out this field”. * “Username”: System display warning message: “please fill out this field”. * “Password”: System display warning message: “please fill out this field”. * “Repeat password”: System display warning message: “please fill out this field”. * “Phone”: System display warning message: “please fill out this field”. | | 4 | Guest input wrong some fields with requirement. | System notices that guest need to re-input all these field:   * “Email”: System display warning message: “Email invalid! ([me@example.com)](mailto:me@example.com))”. * “Email”: System display warning message: “Email must be 10 - 254 characters”. * “Full name”: System display warning message: “Full name must be 10 - 50 characters”. * “Username”: System display warning message: “Username must be 6 - 20 characters”. * “Password”: System display warning message: “Password must be 6 - 12 characters”. * “Repeat password”: System display warning message: “Repeat password does not match password”. * “Phone”: System display warning message: “Phone must be numbers”. |   **Relationships:** N/A  **Business Rules:**   * After registered, information of account will be stored in database of the system with role “User” and status is “active” | | | |

**Table: Register specification**

##### **2.3.1.2. <Guest> Login**

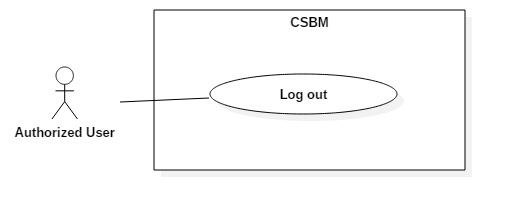


*Figure: <Guest> Login*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC002** | | | |
| **Use Case No.** | 002 | **Use Case Version** | 1.0 |
| **Use Case Name** | Login | | |
| **Author** |  | | |
| **Date** | 25/5/2016 | **Priority** | High |
| **Actor:**   * Guest   **Summary:**   * This use case allows authorized user to log in system.   **Goal:**   * Authentication and authorization.   **Triggers:**   * Guest send the login command.   **Precondition:**   * N/A   **Post Condition:**   * **Success:** Guest will be logged in system with their roles. * **Fail:** Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to login view. | System requires information:   * Username: textbox, required. * Password: textbox, required. | | 2 | Input field. |  | | 3 | Guest sends login command. | Guest will login system with their roles.  [Exception 1] |   **Alternative Scenario:** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest sends login command. | Wrong username or password. Show error message: “Username or password invalid”. |   **Relationships:** N/A  **Business Rules:**   * After login successfully, guest will be redirected to specific view based on their role on the system: user or admin.   + If role is “User”, the system will display to User view.   + If role is “Admin”, the system will display to Admin view. | | | |

**Table: Login specification**

#### **2.3.2. <Authorized User> Overview use case**



*Figure: <Authorized User> Overview Use Case*

##### **2.3.2.1. <Authorized User> Logout**

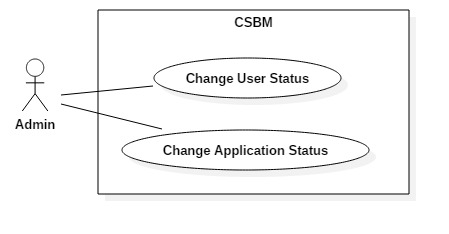


*Figure: <Authorized User> Logout*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC003** | | | |
| **Use Case No.** | 003 | **Use Case Version** | 1.0 |
| **Use Case Name** | Log out | | |
| **Author** |  | | |
| **Date** | 25/5/2016 | **Priority** |  |
| **Actor:**   * Authenticated user.   **Summary:**   * This use case allows Authenticated user logouts the system.   **Goal:**   * User can log out of system.   **Triggers:**   * User sends logout command.   **Precondition:** N/A.  **Post Condition:**   * **Success:** Display login view. * **Fail:** N/A   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click button “Logout” in menu bar. | System clear session state if have, takes user out of system.  System display login view. |   **Alternative Scenario:** N/A  **Exceptions:** N/A  **Relationships:** N/A  **Business Rules:**   * If user logout while send request to system, all request will be delete. * If user is not available longer than 30 minutes, they will see the login view when they be back. | | | |

**Table: Logout specification**

#### **2.3.3. <Admin> Overview use case**



*Figure: <Admin> Overview Use Case*

##### **2.3.3.1. <Admin> Change Application Status**

*Figure: <Admin> Change Application Status*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC004** | | | |
| **Use Case No.** | 004 | **Use Case Version** | 1.0 |
| **Use Case Name** | Change Application Status | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * Admin.   **Summary:**   * This use case allows admin to update application status.   **Goal:**   * Update existed application.   **Triggers:**   * Admin click on “Change status” on an application in “Manage Application” page.   **Precondition:**   * Application exists in storage. * User must log into the system with role admin.   **Post Condition:**   * **Success:** Application status is updated into storage. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin click on “Change status” button. | Confirmation dialog will be shown.  [Alternative 1,2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on “Ok” button. | Maker position will be saved into storage. | | 2 | Click on “Cancel” button. | Confirmation dialog will be canceled and no change is made. |   **Exceptions:** N/A.  **Relationships:** N/A.  **Business Rules:**   * If application’s status is changed to “Deactivate”, the application won’t be able to use and users won’t be able to access the application. * In owner view, users won’t be able to access the application has banner “Deactivate” with red text. * If application’s status is changed to “Activate”, users can access the application. | | | |

**Table: Change Application Status specification**

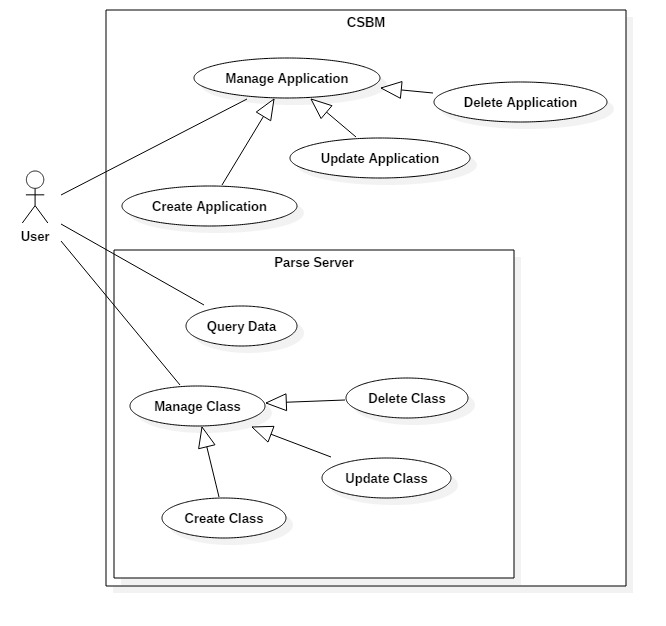
##### **2.3.3.2. <Admin> Change User Account Status**

*Figure: <Admin> Change User Account Status*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC004** | | | |
| **Use Case No.** | 004 | **Use Case Version** | 1.0 |
| **Use Case Name** | Change User Account Status | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * Admin.   **Summary:**   * This use case allows Admin to update user account.   **Goal:**   * Update exists user account.   **Triggers:**   * Admin click on “Change status” on an account in “Manage Application” page.   **Precondition:**   * Account exists in storage. * User must log into the system with role admin.   **Post Condition:**   * **Success:** Account status is updated into storage. * **Fail:** N/A.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin click on “Change status” button. | Confirmation dialog will be shown.  [Alternative 1,2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on “Ok” button. | Maker position will be saved into storage. | | 2 | Click on “Cancel” button. | Confirmation dialog will be canceled and no change is made. |   **Exceptions:** N/A.  **Relationships:** N/A.  **Business Rules:**   * If account’s status is changed to “Deactivate”, the account won’t be able to access the system. * Administrator can’t Deactivate himself. * If account’s status is changed to “Activate”, Guest can use this account to access the system. | | | |

**Table: Change User Status specification**

#### **2.3.4 <User> Overview use case**



*Figure: <User> Overview Use Case*

##### **2.3.4.1. <User> Create New Application**



*Figure: <User> Create New Application*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC008** | | | |
| **Use Case No.** | 008 | **Use Case Version** |  |
| **Use Case Name** | Create New Application | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * User.   **Summary:**   * This use case allows user to create new application.   **Goal:**   * New application will be stored in storage.   **Triggers:**   * User click “Create new application” button on “Manage Application” page.   **Precondition:**   * Actor login to system with role “User”.   **Post Condition:**   * **Success:** All new application’s data were saved successfully. * **Fail:** Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click “Add New Application” button. | System show dialog which contains information of:   * Application Name: textbox, min length 2, max length 10, required. * “Create”: button. | | 2 | Input field. |  | | 3 | Click on “Create” button. | Confirmation dialog will be shown  [Exception 1]  [Alternative 1, 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor action | System Response | | 1 | Click on “Ok” button | New application will be added in storage. | | 2 | Click on “Cancel” button | Confirm dialog will be canceled. A new application isn’t created. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Click on “Create” button | “Application Name” is empty or longer than 10 words. Show error message: “Application Name is limited from 2 to 10 characters”. |   **Relationships:** N/A.  **Business Rules:**   * New Application will be created with active status. * System must ensure has not duplicated application name. * System will notifies owner if the application is added fail. * When user add application from create new application form, system will automatic generate userId by Id’s user create application, random ObjectId, masterKey, generate created\_at and updated\_at by current time and save it to system, for example:  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | ObjectId | masterKey | created\_at | update\_at | name | userId | | 5759360ae77ed0391e143d45 | 38ccZfmdEJXE5axrKh1TTX7Qp3uAfLVjYZYi6mVQFNPi92hn | ISODate("2016-06-09T09:25:30.000Z") | ISODate("2016-06-09T09:25:30.000Z") | test | 5756aaaa6f9268590fa0867b | | | | |

**Table: Create New Application specification**

##### **2.3.4.2. <User> Delete Application**

*Figure: <User> Delete Application*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC009** | | | |
| **Use Case No.** | 009 | **Use Case Version** |  |
| **Use Case Name** | Delete Application | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * User   **Summary:**   * This use case allows user to delete application.   **Goal:**   * Application will be removed from the storage.   **Triggers:**   * User click “Delete Application” button in “Manage Application” page.   **Precondition:**   * Actor login to system with role “User”. * Application exists in the system.   **Post Condition:**   * **Success:** Application will be deleted from storage. * **Fail:** Application is not deleted. Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Application owner click on “Delete Application” button in “Manage Application” page. | System show dialog following information:   * “Application Name”: Dropdown list contain all application name, required. * “Delete”: button. | | 2 | Choose application name. |  | | 3 | Click on “Delete” button. | Confirmation dialog will be shown  [Exception 1]  [Alternative 1, 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor action | System Response | | 1 | Click on “Ok” button | Dialog is turn-off, Application is deleted. | | 2 | Click on “Cancel” button | Dialog is turn-off, Application is not deleted. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor action | System Response | | 1 | Click on “Ok” button | Application is not deleted. Show error message: “Delete application fail. Try again”. |   **Relationships:** N/A.  **Business Rules:**   * Application is still active. | | | |

**Table: Delete application specification**

##### **2.3.4.3. <User> Update Application**

*Figure: <User> Update Application*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC008** | | | |
| **Use Case No.** | 010 | **Use Case Version** |  |
| **Use Case Name** | Update Application | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * User.   **Summary:**   * This use case allows user to update existed application.   **Goal:**   * Update existed application.   **Triggers:**   * User click “Update Application” button in “Manage Application” page.   **Precondition:**   * Actor logged in system with role “User”. * Application exists in the system.   **Post Condition:**   * **Success:** Application will be updated with new information. * **Fail:** Application will not be updated. Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click “Update Application” button on “Manage Application” page. | System show dialog following information:   * “Application Name”: textbox, min length: 1, max length: 100, required. * “Update”: button. * “Close”: button. | | 2 | User fills in information application.  [Alternative 1] |  | | 3 | Click on “Update” button. | Confirmation dialog will be shown.  [Exception 1]  [Alternative 2, 3] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor action | System Response | | 1 | Click on “Close” button. | System close dialog and cancel update process. | | 2 | Click on “Update” button. | Application will be updated. | | 3 | Click on “Cancel” button. | Confirmation dialog will be canceled and no change is made. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Click on “Update” button. | “Application Name” is empty or longer than 100 words. Show error message: “Application Name is limited from 1 to 100 characters”. |   **Relationships:** N/A.  **Business Rules:**   * New information of application will be stored in storage. * System must ensure has not duplicated application name. | | | |

**Table: Update Application specification**

##### **2.3.4.4. <User> Update Class Field**

*Figure: <User> Update Class Field*

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC008** | | | |
| **Use Case No.** | 011 | **Use Case Version** |  |
| **Use Case Name** | Update Class Field | | |
| **Author** |  | | |
| **Date** | 25/05/2016 | **Priority** |  |
| **Actor:**   * User.   **Summary:**   * This use case allows user to update existed class.   **Goal:**   * Update existed class.   **Triggers:**   * User click “Change Field” button in “Manage Application” page.   **Precondition:**   * Actor logged in system with role “User”. * Class Field exists in the system.   **Post Condition:**   * **Success:** Class field will be updated with new information. * **Fail:** Class field will not be updated. Error message will be shown.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click “Change Field” button on “Manage Application” page. | System show dialog following information:   * “Field Name”: Dropdown list contain all class field name, required. * “New Field Name”: textbox, min length: 1, max length: 100, required. * “Update”: button. * “Close”: button. | | 2 | User fills in information application.  [Alternative 1] |  | | 3 | Click on “Update” button. | Confirmation dialog will be shown.  [Exception 1]  [Alternative 2, 3] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor action | System Response | | 1 | Click on “Close” button. | System close dialog and cancel update process. | | 2 | Click on “Update” button. | Application will be updated. | | 3 | Click on “Cancel” button. | Confirmation dialog will be canceled and no change is made. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Click on “Update” button. | “New Field Name” is empty or longer than 100 words. Show error message: “Field Name is limited from 1 to 100 characters”. |   **Relationships:** N/A.  **Business Rules:**   * New information of field will be stored in storage. * System must ensure has not duplicated field name in available class. | | | |

**Table: Update Application specification**

## **Software System Attribute**

### **3.1. Usability**

#### **3.1.1. Graphic user interface**

All the test, labels and alerts of web application and mobile framework will be written by English.

#### **3.1.2. Usability**

The system usability is easy to use that will need less than 1 hour for training to use system.

#### **3.1.3. Installation**

User can follow installation and manual guide for installation. If there are any problems, user cans contacts developer for help.

### **3.2. Reliability**

* The data should be backed up everyday.

### **3.3. Availability**

* The service relates to communication so it can be available 24/7.
* Server should have back-up method to make sure that if it having problems, all necessary data can be protected and restore easily.

### **3.4. Security**

* Input data is validated if necessary before saving to database.
* Users is authenticated/authorized for all users when they logged in to the system.

### **3.5. Maintainability**

* System is separated into modules.

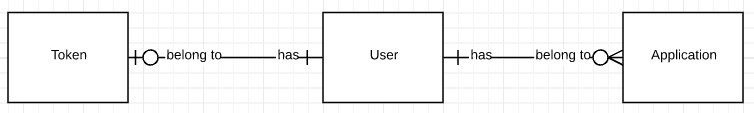
### **3.6. Portability**

* User, guest can use application on every OS supported web browsers.
* User can use mobile framework for develop any Android or iOS application.

### **3.7. Performance**

* Requests from web application are responded in less than 10 seconds at 8 Mbps bandwidth speed.

## **Conceptual Diagram**



**Figure: Conceptual Diagram**

|  |  |
| --- | --- |
| **Entity Data dictionary: describe all content of all entities** | |
| **Entity Name** | **Description** |
| User | Contain the user information |
| Token | Contain the token information |
| Application | Contain the application information |

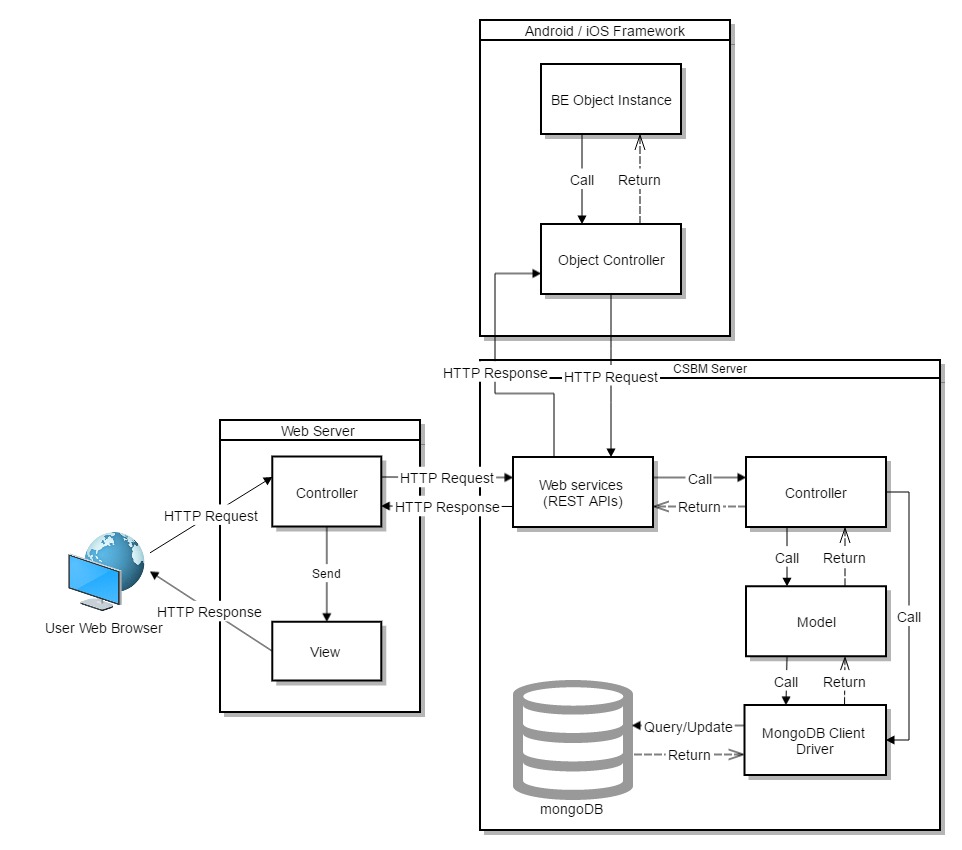
**Table: Conceptual diagram data dictionary**

# **D. Software Design Description**

## **Design Overview**

* This document describes the technical and user interface design of **CSBM System**. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.
* The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.
* The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.
* The database design describes the relationships between entities and details of each entity.
* Document overview:
  + - Section 2: gives an overall description of the system architecture design.
    - Section 3: gives component diagrams that describe the connection and integration of the system.
    - Section 4: gives the detail design description, which includes class diagram, class explanation, and sequence diagram to details the application functions.
    - Section 5: describe screens design.
    - Section 6: describe a fully attributed ERD.
    - Section 7: describe algorithms.

## **System Architectural Design**



**Figure: System architecture design**

* 1. **Web application architecture description**

In Web Application, the system is developed under AngularJS MVC architecture style. We choose this architecture for Web application because of following advantages:

* Web app contains a Web service with MVC architecture, we can separate business code with Controller and View, so we can use the business code in web service without repeat the code.
* We can organize the code better for maintainability, extensibility, reusability.
* In scope of 4-member team, MVC architecture make it easier to split the big project into small modules and make it easier to assign each module for members in our team.

The project follows MVC architecture with following components:

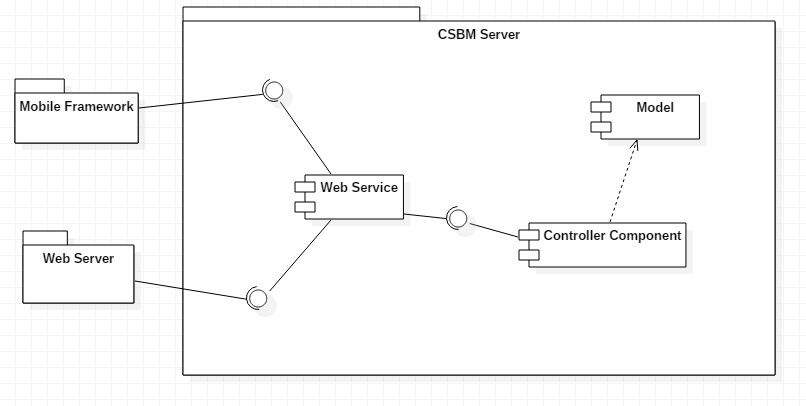
* **Controller** is the parts of the controller that acts like event handler to handles user interaction.
* **View** is the parts of the web server that acts like event handles the display of the data. The selection of View is under control of Controller.
  1. **CSBM application architecture description**

The application is developed under MVC architecture style.

This project follows MVC architecture with following component:

* **Web services** is the parts of the application that acts like event handler for web and mobile communication via REST method.
* **Controller** is the parts of the application that read data from a request and calls appropriate MongoDB Client Driver’s method then return to web services.
* **Model** is the parts of the application that acts like data transfer object between the system and database.
* **Web services** is the parts of the application that acts like event handler for web and mobile communication via REST method.
* **MongoDB Client Driver** is the parts of the application that provides some methods and classes to persistent store data into database.
  1. **Mobile framework architecture description**

1. **Component Diagram**

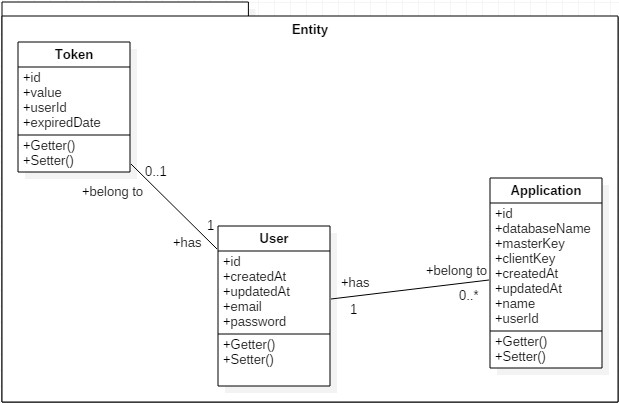
****

**Figure: Component Diagram**

|  |  |
| --- | --- |
| Component Dictionary: Describes components | |
| Mobile Framework | Mobile Framework package |
| Web Server | Web Server package: View, Controller |
| Web Service | Include all web API Controllers of the system |
| Controller Component | Component is used to handle request |
| Model |  |

**Table: Component Dictionary**

1. **Detail Description**
   1. **Class Diagram**

****

**Figure: Class Diagram**

|  |  |  |
| --- | --- | --- |
| Class dictionary: describe Class | | |
| Class Name | **Mapping column with Conceptual diagram** | **Description** |
| User | User | Contain the user information |
| Token | Token | Contain the token information |
| Application | Application | Contain the application information |

**Table: Class Dictionary**

* 1. **Class Diagram Explanation**
     1. **User**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **id** | String | Private | Unique identifier of a user |
| **createdAt** | Date | Private | Date of create user |
| **updatedAt** | Date | Private | Date of update user |
| **email** | String | Private | User’s email |
| **password** | String | Private | User’s password |

**Table: Account Attributes**

* + 1. **Token**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **id** | String | Private | Unique identifier of a token |
| **value** | String | Private | Token’s value |
| **userId** | String | Private | User’s Id |
| **expiredDate** | Date | Private | Token’s expiredDate of license |

**Table: Token Attributes**

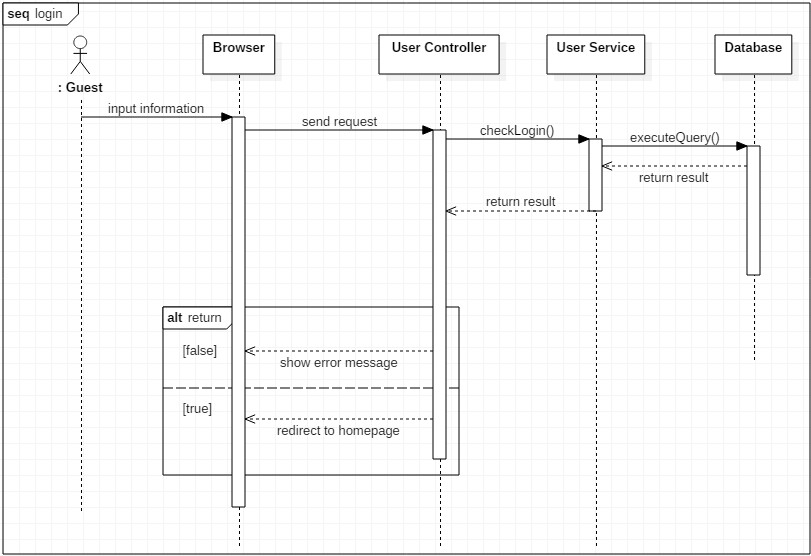
* + 1. **Application**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| **id** | String | Private | Unique identifier of an application |
| **databaseName** | String | Private | Unique identifier of an application database |
| **masterKey** | String | Private | Application’s MasterKey |
| **clientKey** | String | Private | Application’s ClientKey |
| **createdAt** | Date | Private | Date of create application |
| **updatedAt** | Date | Private | Date of update application |
| **name** | String | Private | Application’s Name |
| **userId** | String | Private | User’s Id |

**Table: Application Attributes**

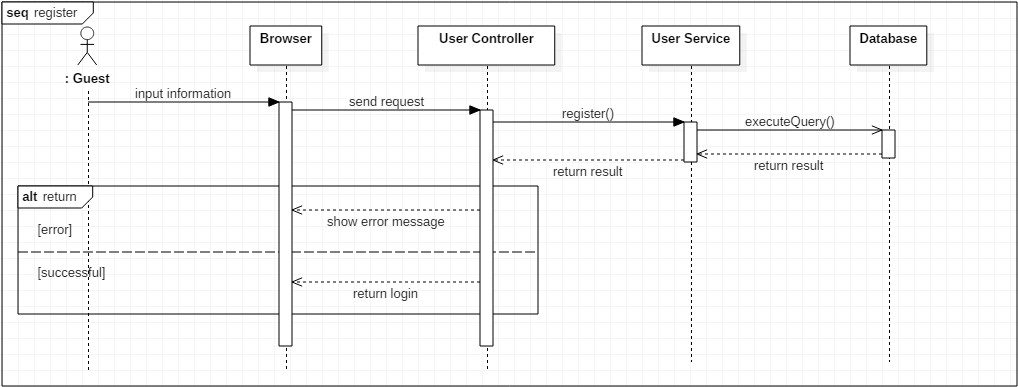
* 1. **Interactive Diagram**
     1. **Web Application**
        1. **<Guest>**
           1. **Login**

Summary: This diagram show how a guest login system.

**Figure: Sequence Diagram - <Guest> Login**

* + - * 1. **Register**

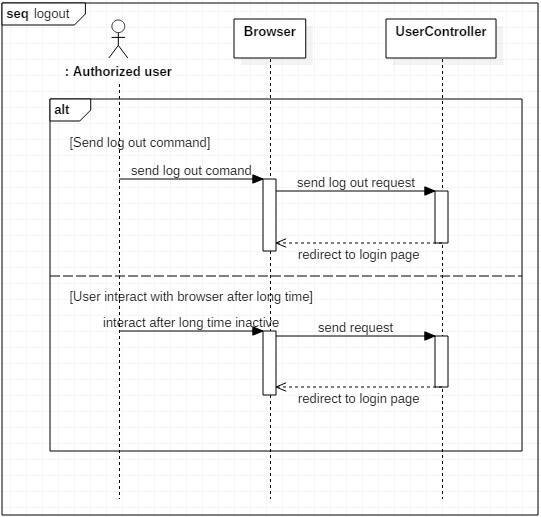
Summary: This diagram show process of register



**Figure: Sequence Diagram - <Guest> Register**

* + - 1. **<Authorized User>**
         1. **Logout**

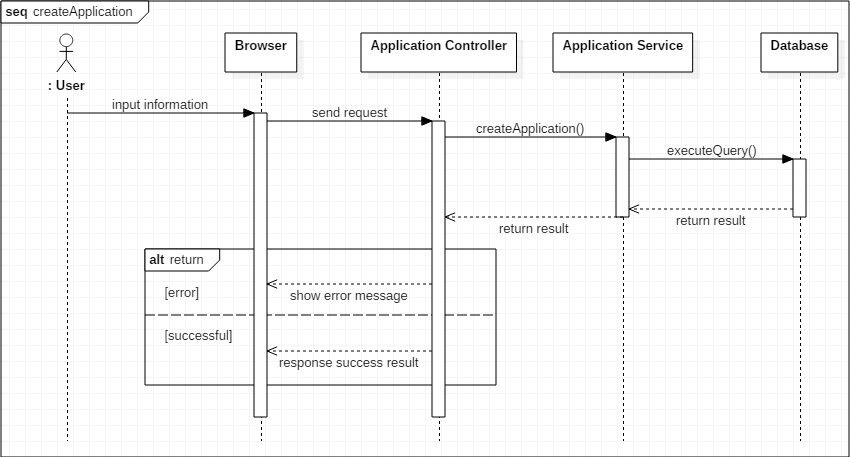
Summary: This diagram show process of Logout

****

**Figure: Sequence Diagram - <Authorized User> Logout**

* + - 1. **<User>**
         1. **Create New Application**

Summary: This diagram show process of Create New Application



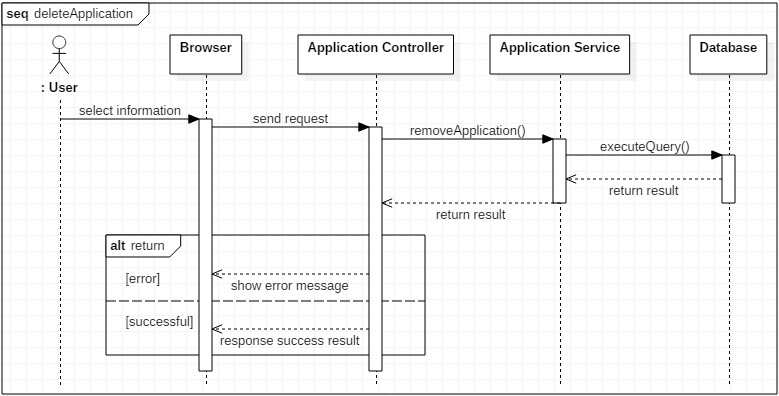
**Figure: Sequence Diagram - <User> Create New Application**

* + - * 1. **Update Application**

Summary: This diagram show process of Update Application

* + - * 1. **Delete Application**

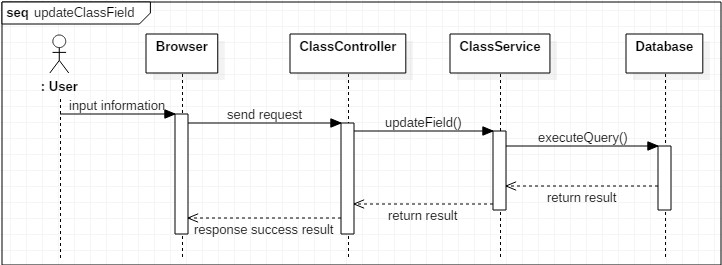
Summary: This diagram show process of Delete Application

****

**Figure: Sequence Diagram - <User> Delete Application**

* + - * 1. **Update Class Field**

Summary: This diagram show process of Update Class Field

****

**Figure: Sequence Diagram - <User> Update Class Field**

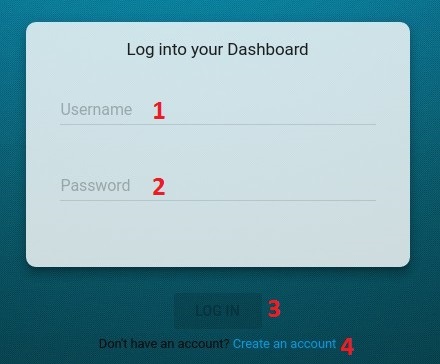
* + - 1. **<Admin>**
         1. **Change Application Status**

Summary: This diagram show process of Change Application Status

* + - * 1. **Change User Status**

Summary: This diagram show process of Change User Status

1. **Interface**
   1. **Component Interface**
   2. **Web Application Design**
      1. **Guest**
         1. **Login**

****

**Figure: Web Application Desgin - <Guest> Login**

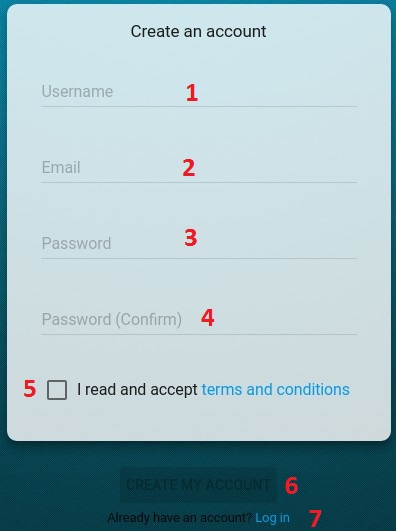
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | txtUsername | Fill username | No | Yes | Textbox | String | 6 - 20 |
| 2 | txtPassword | Fill password | No | Yes | Textbox | String | 6 - 12 |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | btnLogin | Send login command | Validation username and password fields | Hone screen |
| 4 | linkToRegister | View register page | N/A | Transfer to register page |

* + - 1. **Register**

****

**Figure: Web Application Desgin - <Guest> Register**

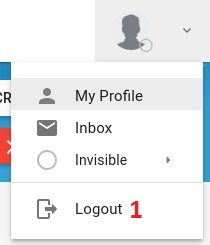
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | txtUsername | Fill username | No | Yes | Textbox | String | 6 - 20 |
| 2 | txtEmail | Fill email | No | Yes | Textbox | String | 10 - 254 |
| 3 | txtPassword | Fill password | No | Yes | Textbox | String | 6 – 12 |
| 4 | txtPasswordConfirm | Fill password confirm | No | Yes | Textbox | String | 6 - 12 |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | btnAccept | Accept terms and conditions | N/A | N/A |
| 6 | btnRegister | Send register command | Validation username and password fields | Transfer to login page |
| 7 | linkToLogin | View login page | N/A | Transfer to login page |

* + 1. **Authorized User**
       1. **Logout**

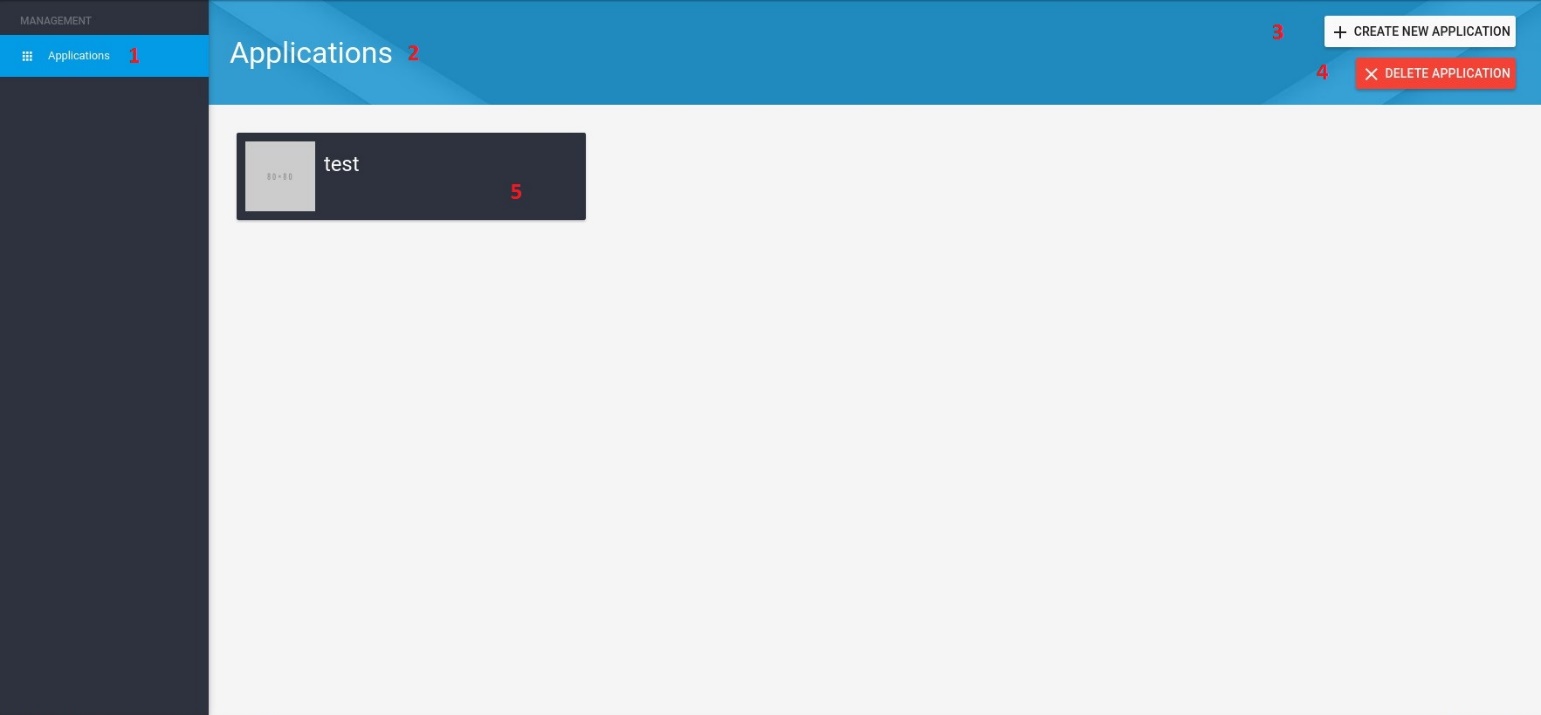
****

**Figure: Web Application - <Authorized User> Logout**

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | btnLogout | Send log out command | N/A | Destroy session and redirect to login page |

* + 1. **User**
       1. **Manage Applications**



**Figure: Web Application Design - <User> Manage Applications**

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Menu | Navigation bar | Yes | Yes | Menu bar | N/A | N/A |
| 2 | Title | Title of the page | Yes | Yes | Label | N/A | N/A |
| 5 | Application Card | List Applications | Yes | Yes | Card | N/A | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | btnCreateDialog | Create application dialog button | N/A | Show create application dialog |
| 4 | btnDeleteDialog | Delete application dialog button | N/A | Show delete application dialog |

* + - 1. **Create Application**

****

**Figure: Web Application Design - <User> Create Application**

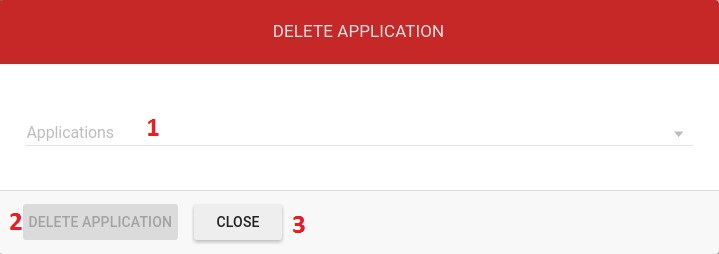
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | txtApplicationName | Fill application name | No | Yes | Textbox | String | 1 - 12 |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | btnCreateApp | Send create application command | N/A | Submit create application |

* + - 1. **Delete Application**

****

**Figure: Web Application Design - <User> Delete Application**

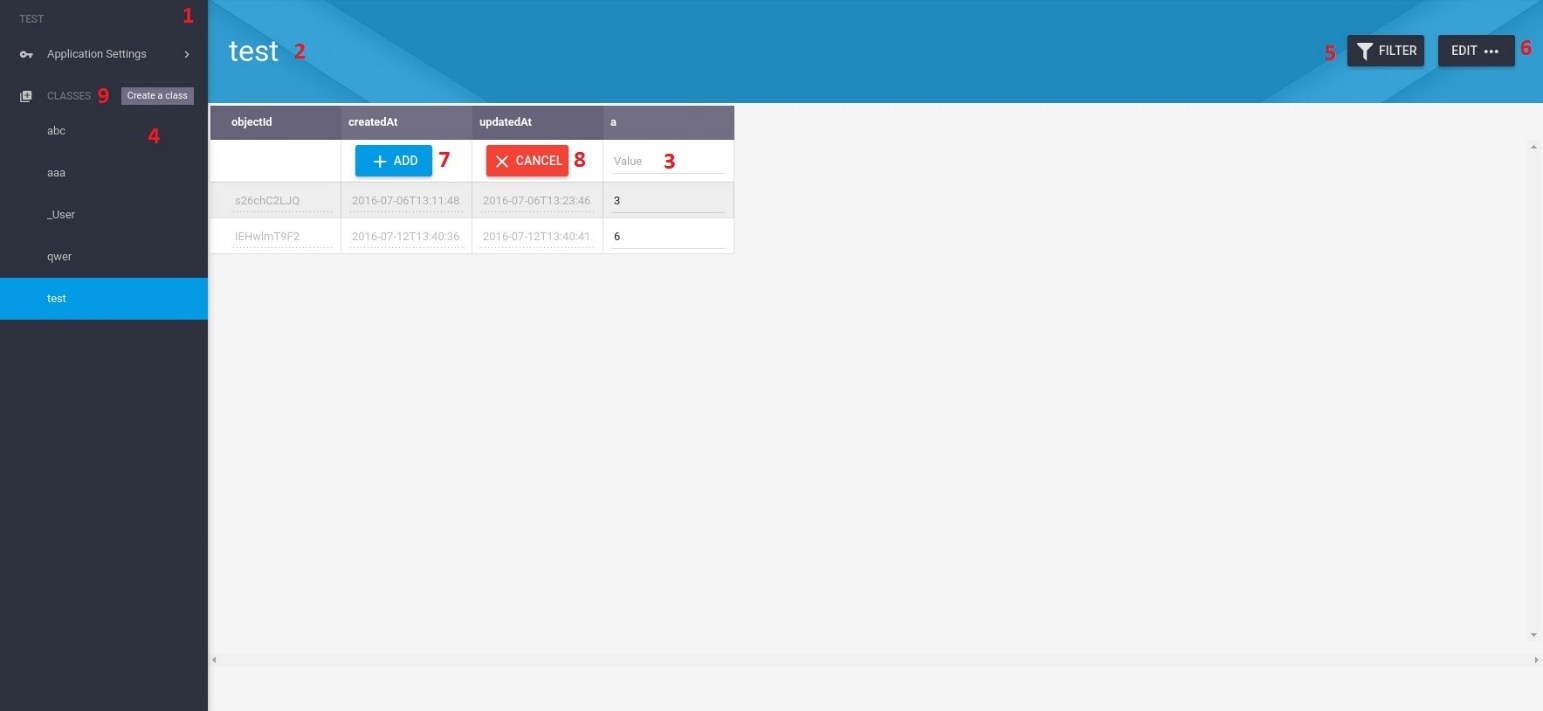
**Fields**

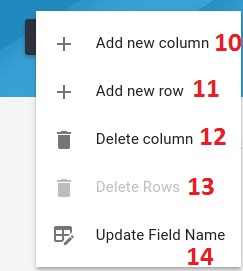
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | optApplications | Application will be delete | No | Yes | Select box | Integer | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | btnDeleteApp | Send delete application command | Validate all required fields before submit | Submit delete application |
| 3 | btnCloeDialog | Close dialog button | N/A | Close delete application dialog |

* + - 1. **Manage Classes**

****

****

**Figure: Web Application Design - <User> Manage Classes**

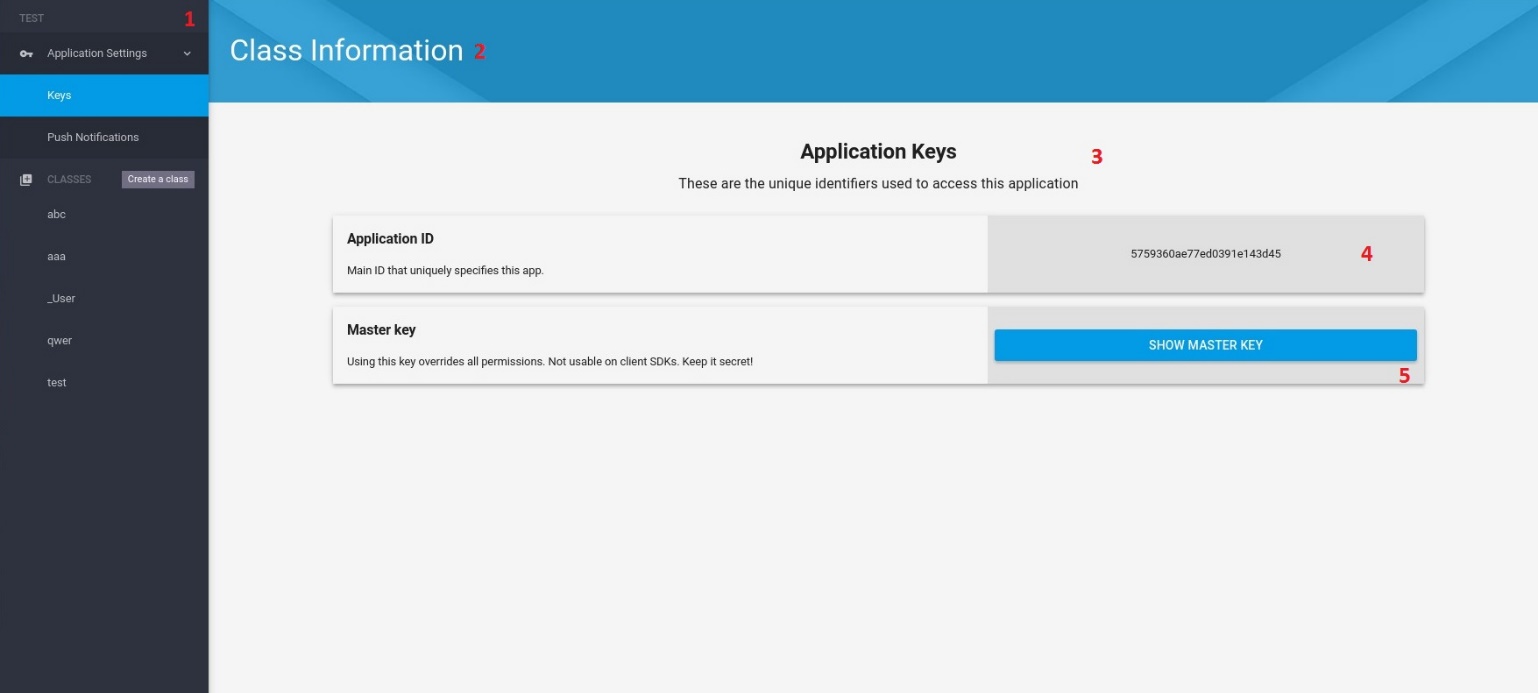
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Menu | Navigation bar | Yes | Yes | Menu bar | N/A | N/A |
| 2 | Title | Title of the page | Yes | Yes | Label | N/A | N/A |
| 3 | txtValue | Fields Value | No | No | Textbox | String | 0 – 254 |
| 4 | Classes Menu | Navigation bar | Yes | Yes | Menu bar | N/A | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | btnFilterDialog | Filter dialog button | N/A | Show filter dialog |
| 6 | btnEdit | List action button | N/A | Show list Action |
| 7 | btnAdd | Send add new schema command | N/A | Add new row to list data |
| 8 | btnCancel | Cancel add new data to list button | N/A | Hide add new row to list data |
| 9 | btnCreateClassDialog | Create new class dialog button | N/A | Show create new class dialog |
| 10 | btnAddColumnDialog | Add new field to list dialog button | N/A | Show add new column to table dialog |
| 11 | btnAddNewRow | Add a new data to list button | N/A | Show information row to add new |
| 12 | btnDeleteClumnDialog | Delete exists field list dialog button | N/A | Show delete column dialog |
| 13 | btnDeleteRow | Send delete schema(s) command | N/A | Delete schema(s) is selected in list data |
| 14 | btnUpdateFieldDialog | Update field name dialog button | N/A | Show update field name dialog |

* + - 1. **Application Key**

****

**Figure: Web Application Design - <User> Application Key**

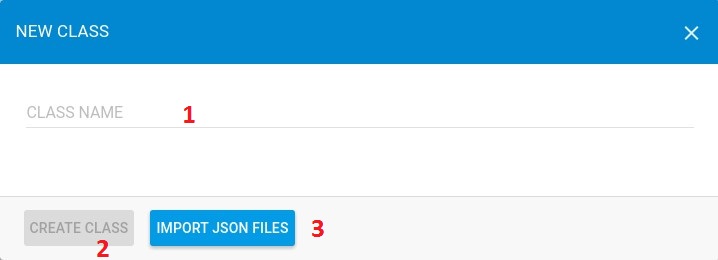
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Menu | Navigation bar | Yes | Yes | Menu bar | N/A | N/A |
| 2 | Title | Title of the page | Yes | Yes | Label | N/A | N/A |
| 3 | lbDescription | Description of application key | Yes | Yes | Label | N/A | N/A |
| 4 | lbApplicationId | Application Id | Yes | Yes | Label | N/A | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 5 | btnMasterKey | Send get application master key command | N/A | Show application master key |

* + - 1. **Push Notification**
      2. **Create Class**



**Figure: Web Application Design - <User> Create Class**

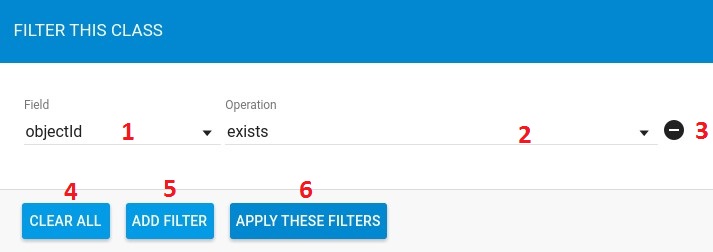
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | txtClassName | Fill class name | No | Yes | Textbox | String | 1 - 12 |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | btnCreateClass | Send create class command | N/A | Submit create class |
| 3 | btnImportFileDialog | Import file dialog button | N/A | Show import file dialog |

* + - 1. **Filter Class Fields**



**Figure: Web Application Design - <User> Filter Class Fields**

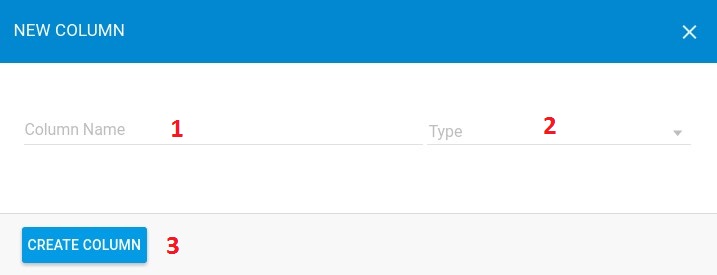
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | optFields | Fields will be filter | No | Yes | Select box | Integer | N/A |
| 2 | optOperation | Operation will be use to filter | No | Yes | Select box | Integer | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | btnRemoveFilter | Remove filter option button | N/A | Filter option will be remove |
| 4 | btnClearAll | Remove all filter option button | N/A | All Filter option will be remove |
| 5 | btnAddFilter | Add new filter option button | N/A | Add new filter option |
| 6 | btnApply | Apply filter button | N/A | Apply all fiter option to list data |

* + - 1. **Add New Column**



**Figure: Web Application Design - <User> Add New Column**

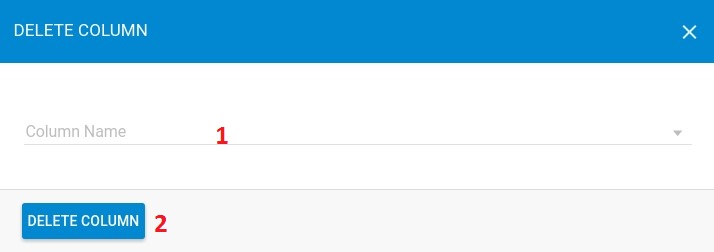
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | txtColName | Fill column name | No | Yes | Textbox | String | 1 - 12 |
| 2 | optType | Select column type | No | Yes | Select box | Integer | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | btnCreateCol | Send create new field command | N/A | Submit create column |

* + - 1. **Delete Column**



**Figure: Web Application Design - <User> Delete Column**

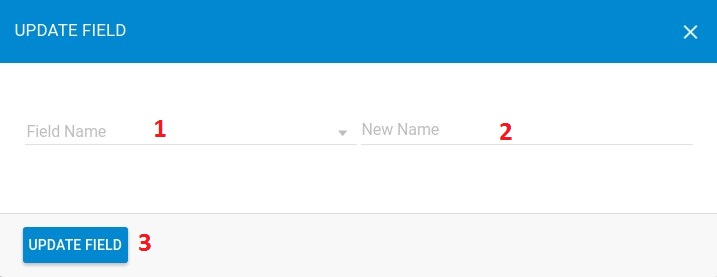
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | optColName | Select Column Name | No | Yes | Select box | Integer | N/A |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | btnDeleteCol | Send delete field command | N/A | Submit delete column |

* + - 1. **Update Field Name**



**Figure: Web Application Design - <User> Update Field Name**

**Fields**

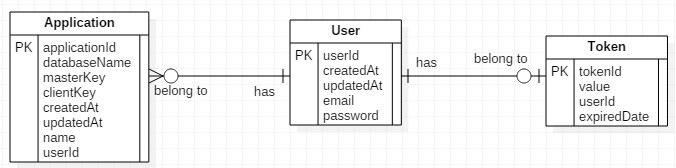
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read Only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | optFieldName | Select field name to update | No | Yes | Select box | Integer | N/A |
| 2 | txtFieldNewName | Fill field new name | No | Yes | Textbox | String | 1 - 12 |

**Buttons/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | btnUpdateField | Send update field command | N/A | Submit update field command |

* + 1. **Admin**
       1. **Change Application Status**
       2. **Change User Status**
  1. **Mobile Application Design**

1. **Database Design**
   1. **Entity Relationship Diagram**

****

**Figure: Entity relationship diagram**

* 1. **Entity Dictionary**

|  |  |  |
| --- | --- | --- |
| **Entity Data Dictionary: describe content of all entities** | | |
| **Entity name** | **Description** | **Mapping column with Conceptual diagram** |
| **User** | Contain the user information | user |
| **Application** | Contain the application information | application |
| **Token** | Contain the token information | token |

**Table: Entity dictionary**

1. **Algorithms**