**Contents**

**Abstract 1**

**Acknowledgement 2**

**List of Figures**

**1 Introduction**…………………………………………………………………………………….. **6**

1.1 Background and motivations……………………….…… 7

1.2 Objectives………………………………………………….. 7

1.3 Scope……………………………………………………….. 8

**2 Literature Review**…………………………………………………………………………. **9**

2.1 What is Literature Review ………………………………………… 9

2.2 Related Works ………………………………………………………..9

**3 System Description ………………………………………………10**

3.1 Existing System …………………………………………………….10

3.2 System Proposal ……………………………………………………..10

3.2.1 Server Site ……………………………………………..11

3.2.2 Client Site ………………………………………………….11

**4 Requirement Analysis…………………………11**

4.1 What is the Requirement Analysis? ………...12

4.1.1 Functional Requirement…………12

4.1.2 Non-functional Requirements …………….12

4.1.3 Domain Requirements …………..12

4.2 Functionalities of the System ………………………..13

4.2.1 API Admin Woks ……………………..….13

4.2.2 Developer Works ………….………….……13

4.2.3 User Works ……………………………….13

4.3 Required Tools and Technologies ……………………………14

4.3.1 Spring Boot ………………………………………...14

4.3.2 MySQL …………………...15

4.3.3 Bootstrap ……………………...16

4.3.4 Thyme leaf ………………………….……………....16

4.3.5 HTML ……………………...17

4.3.6 CSS.………………………….……...17

4.3.7 Hibernate ……………………………….………...18

4.3.8 Java ….……………………………...18

**5 System Analysis**

5.1 Why needed? ………………………………………….19

5.2 System Analyst ………………………………………………….19

5.3 Basic Task of System Analyst …………. 20

5.4 Procedural Steps of System Analysis….…………….20

5.5 Techniques and Tools for Requirements Collecting……..20

5.6 User of the System. …………………………………. 21

5.7 System Planning ………………………………. 21

5.7.1 Feasibility Study …………………………….21

5.7.2 Work Plan………………..………………….22

5.8 Adopted Analysis Strategy…………………………….22

**6 System Design………………………………………..22**

6.1 Design Necessity ………………………………………….23

6.2 Database Design ………………………….23

6.2.1 Database Design Process……………………….24

6.2.2 Database Table……………………………….24

6.3 Object Oriented Design…………………….24

6.3.1 Context Diagram ……………………………25

6.3.2 Use Case Design ……………………………………25

6.3.3 Class Diagram…………………………………….26

6.3.4 Sequence Diagram………………………………….28

6.5 User Interface Design……………………………………… 30

6.5.1 Design Flexibility.……………………….30

6.6 Programming Patterns. ………………………30

6.6.1 MVT Patterns …..…………………30

6.6.2 Component Interaction Regarding the Analysis…..31

**7 System Development………………………….31**

7.1 Why perform System Development? …….…………………32

7.2 Development Strategy ……………32

7.3 System Environment ……………………………32

7.4 Development Strategy ………………………………33

7.5 System Development Life Cycle……..………………33

7.6 Implementation of System ………………………36

7.6.1 Spring Boot Framework …….…………36

7.6.2 Goals of Spring Boot………….………………36

7.6.3 How does it work? ……………..…………37

7.6.4 Spring Boot Starter…………………….37

7.6.4 Advantages of Spring Boot……………………39

**8 Testing and Maintenance……………………………….39**

8.1 Testing and maintenance of the System ………..39

8.2 System Testing ….………………………………………..40

8.2.1 System Testing Activities ………………..40

8.2.2 Necessity of System Testing …………………..40

8.3 Maintenance………………………41

8.4 Different Types of Maintenance………………….41

8.4.1 Corrective Maintenance………………………….41

8.4.2 Adaptive Maintenance………………………41

8.4.3 Perfective Maintenance…………………….42

8.5 Necessity of Maintenance ……………….42

**9 Result and Discussion…………………..43**

9.1 Result and Discussion of the System ……………..43

9.1.1 Result ………………………43

9.1.2 Discussion ……………………..45

**10 Conclusion & Future Work …………………..43**

10.1 Project Benefits ……………..43

10.2 Future Work ………………………43

10.3 Personal Achievement ……………………..45

10.3 Conclusion ……………………..45

**Related Works**

**References**

**List of Figures**

1. System Design …………………………
2. Context Diagram………………………………………..
3. Use Case Diagram………………………………………………….
4. Sequence Diagram 1……………………………………
5. Sequence Diagram 2……………………………
6. Sequence Diagram 3……………………………
7. Sequence Diagram 4……………………………………
8. Component Interaction within MVT Patterns…………………
9. Testing Strategy……………

**Abstract**

There are various functional process involved to operate a web application effectively. This project’s functional process involved in designing and development of a generalized add-on library module or package for web application. Main purpose is to build such an API that will provides a user defined information system with a great interface.

Despite of having difference in functional purpose of many web application; web app can contains similar task of information storing and finding efficiently based of the web app’s purpose. But unfortunately we don’t have any. Actual motive is to solve the problem by creating an API which can be used almost every web application as a functional component. Spring boot framework technology is used to develop the API.

This generalized add-on library package can easily access the web app component and deliver the services with more flexibility. It will result an excellent API which is capable of providing a dynamicity in information managing with user interface. As this is a library package it will be used as a component and can save a lot of time and work. It help to open access in faster method than traditional publishing.

**Introduction**

An API can be viewed as a public contract because the author defines how it must be used and what it will return without necessarily exposing how the functionality is implemented. One of the most powerful features of any library or service is the ability to be integrated with other applications. An API facilitates these types of integrations by defining public interfaces which can be easily used.

The interfaces mentioned can be of different types. One way to use a library could be by using a class, static methods, or simply sending a message to a centralized message queue. In our library package, we use static methods. Managing all information is an essential part of any organization .Our project takes the concern of making a standard library package in an automated way.

In every organizations, all of the functional components, it is needed to collect essential records .And it takes much time and effort of the staffs of the organizations to collect information repeat about the same persons for their needs. If all the information can be recorded automatically in a computerized way it becomes much helpful, less complicated and time saving for all the staffs concerned.

**Background and Motivation**

Removing redundancy has been one of the oldest challenge for computer science since computer was introduced. Information management system is an important part for almost every web application and here comes the question of static work. Developers need to create their own information management system but having an automated system can solve the problem easily.

So introducing dynamicity in information management to several web application is the reason for the developing this API. Replacing static work with dynamic can save the lot of work and time. In will a new dimension in the field of development and research. This generalized library package will provide ease to access the web app component and deliver the services with more reliability. This can improve a lot of web app that uses information system management as component. Additionally, it gives an ID card generation feature for specific type of information management system.

**Objectives**

Project objectives are:-

* Create an API for generalized use for developer.
* Handling dynamic information management system.
* Find information record by QR code.
* Generate ID card for specific field.
* User authentication with spring security.

**Scope**

As library package is nothing more than a container of compiled programs it can be used any web app by importing.

* Development purpose: Web application that needs to have a function of information management system can use the library and save their lot of work.
* Research purpose: Can be useful for research like medicine, technology, economy etc.
* Educational purpose: Certain educational institute can use this library package to manage information like:

1. University Information Management System.
2. Department Information Management System.
3. Teachers Information Management System.
4. Student Information Management System.
5. Library Information Management System.
6. Staff Information Management System.

* Organizational purpose: Almost every organization needs an information base of their employee and this can be easily done through this API. Some example fields can be:

1. Employee Information management system.
2. Doctor Information management system.
3. Transection Information management system;
4. Product information management system.

**Literature Review**

Literature review is the related works that has been done in this field. This can be said as the related works to the field.

**What is Literature review?**

Literature review is the related works to the project concerned. These are the works those are done before this project was initiated. Literature reviews are the previous works that inspired and helped modeling the current project with many tips .As much as related works are reviewed, there is chance of less failure and less time consumption. And after implementation less complaints may be expected.

**Related Works**

**System Description**

System description includes the description of the system as it develops. It is necessarily going to specify how the system is how the system is going to be and how will be work from the perspective of the server and clients.

**Existing System**

The existing system of making a list for individual system in a conventional way and it takes much time. It has the following features:-

* Need to make individual form to collect information for individual project or functional components.
* Authentication is needed for any data collection.
* Process becomes slow.
* It cannot be integrated with other application system.
* Sometimes, records are uncertain.
* Unnecessary waste of time.
* It arises in big organization system.

**System Proposal**

The system is built with a two way process of server and client site. System includes static classes for dynamic information management. It’s almost developer and server site system. System work as interface between client and implemented system. Two performed system site are describing follow:

* **Server Site**

Server site is controlled by API which contains the interface for information management system and related task. After integrating the API to developer’s web app server site will provides the feature for the client directly. In this providing it renders the authentication page for user to registration. After sing up it will provide the developer to create their own form and generate a link for users of that project.

* **Client Site**

In this project our client site will be the developer. Developer will integrate the library to their project. After integrating the library developer can access the API server and create his own form by providing the information needed for creating a form like form name and number of field. After that developer will provide the filed type for every filed. Then the form will be created for the developer to use.

**Requirement analysis**

The chapter introduces to the requirements analysis, which is the basic for building the system. According to the requirements specified the requirement of tools and Technologies are decided.

**What is the Requirement Analysis?**

The description of the services and constraints are the requirements for the system. Requirements are the basics for any system building. Requirements need to be very clear and a clear understanding of the requirements make the project successful. System's services, constraints and goals are established consulting with the system users, they are defined in details and used as a system specification.

Software system requirements are:

1. Functional system requirement.
2. Non-functional system requirement.
3. Domain requirements.

**Functional Requirements**

These are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations.

**Non-Functional Requirements**

These are constraints on the services or functions offered by the system, e.g., timing constraints.

**Domain Requirements**

These requirements come from the application domain of the system and that reflect characteristics of that domain. They may be functional or non-functional requirements.

**Functionalities of the system**

The system has 3 actor.

* API Admin
* Developer
* Integrated web app’s user

**Functions performed by API Admin are:**

* Providing interface for system access.
* Develop dynamic form management.
* Add authentication system to the system.
* Create default ID card generation technique for specific field.
* Arrange record using QR code.
* Create QR based search system for records.

**Functions performed by Developers are:**

* Integrate library to required web application.
* Perform registration for form generation.
* Create custom form for information system.
* Can generate ID card if it’s permitted by API.
* Update form according to purpose.
* Perform CRUD operation into integrated system’s user data.

**Functions performed by User are:**

* Fill up form.
* Can update only his/her own record.

**Required Tools and Technologies**

To develop the project, following tools and technologies are used here:

* Spring Boot
* MySQL
* Bootstrap
* Thymeleaf
* HTML
* CSS
* Hibernate
* Java

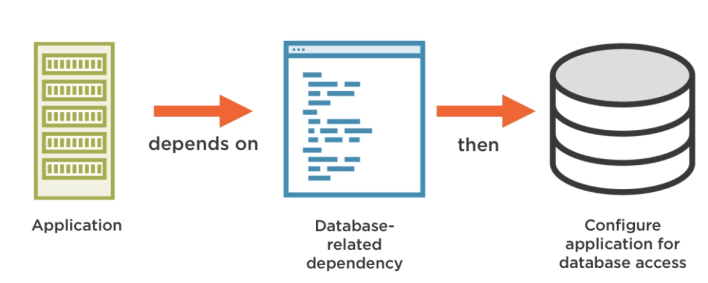
**Spring Boot**

Spring Boot is a project built on the top of the spring framework. It provides a simpler and faster way to set up, configure, and run both simple and web-based applications.

**Notable features:**

Auto-configuration: It sets up your application based on the surrounding environment, as well as hints what the developers provide.

Standalone: Literally, it's completely standalone. Hence, you don’t need to deploy your application to a web server or any special environment. Your only task is to click on the button or give out the run command, and it will start. Opinionated: This means that the framework chooses how to things for itself.



Let’s see an example according to a database. If we add a dependency to the pom.xml, which relates to a database, the framework assumes

That we probably would like to use a database. Then, it auto-configures our application for database access. Furthermore, if the dependency appears for a very specific database, for example, Oracle or MySQL. It can make a more certain assumption and probably will configure that specific database access what we exactly need.

**MySQL**

MySQL is an Oracle-backed open source relational database management system ([RDBMS](https://searchdatamanagement.techtarget.com/definition/RDBMS-relational-database-management-system)) based on Structured Query Language ([SQL](https://searchsqlserver.techtarget.com/definition/SQL)). MySQL runs on virtually all platforms, including [Linux](https://searchdatacenter.techtarget.com/definition/Linux-operating-system), UNIX and [Windows](https://searchwindowsserver.techtarget.com/definition/Windows). Although it can be used in a wide range of applications, MySQL is most often associated with web applications and online publishing.

1. MySQL is written in C and C++ and accessible.
2. MySQL uses an access privilege and encrypted password system that enables host-based verification.
3. MySQL clients can connect to MySQL Server using several protocols, including TCP/IP sockets on any platform.
4. MySQL also supports a number of client and utility programs, command-line programs and administration tools such as MySQL Workbench.

**Bootstrap**

Bootstrap is a free and open-source front-end library for designing websites and web applications. It contains (HTML and CSS)-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

Source: Bootstrap (front-end-framework) (Bootstrap (front-end-framework), i.e.)

**Thymeleaf**

Thyme leaf is a Java XML/XHTML/HTML5 template engine that can work both in web and non-web environments. It is better suited for serving XHTML/HTML5 at the view layer of MVC-based web applications, but it can process any XML file even in offline environments. It provides full Spring Framework integration. Theme Leaf develops free and premium Twitter Bootstrap themes and website templates for general business applications. The Leaf theme is a stylish and customizable Word Press theme. Make it ours with the easy customizable theme options. Uses Super fish menu effects, built-in pagination for post pages, special styles for eight different post formats, and a responsive layout for mobile devices.

**HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications .With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web (WWW).Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages.

HTML describes the structures of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages .With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page.

It provides a means to create structured documents by denoting structural semantics for text such as headings paragraphs, lists, links, quotes and other items.HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input/> introduce content into the page directly. Others such as <p/> surround and provide information about document text and may include others tags as sub-elements. Browsers don not display the HTML tags, but use them to interpret the content of the page.

Source: HTML (HTML, n. d.)

**CSS**

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation of a document in HTML or XML dialects such as (SVG or XHTML).CSS describes how elements should be rendered on screen, on paper, in speech or on other media.CSS is one of the core of the open web and is standardized across browsers according to the W3C specification.

**What is CSS?**

* CSS stands for Cascading Style Sheets
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media
* CSS saves a lot of work. It can control the layout of multiple web pages all at once
* External style sheets are stored in CSS file.

Source: CSS (CSS, n. d.)

**Hibernate**

Hibernate is an open source object relational mapping ([ORM](https://searchwindevelopment.techtarget.com/definition/object-relational-mapping)) tool that provides a [framework](https://whatis.techtarget.com/definition/framework) to map [object-oriented](https://searchmicroservices.techtarget.com/definition/object-oriented-programming-OOP) domain models to relational databases for web applications.

Hibernate ORM is an object-relational mapping tool for the Java programming language. It provides a framework for mapping an object-oriented domain model to a relational database. The Hibernate ORM framework guides mapping [Java](https://www.theserverside.com/definition/Java) [classes](https://whatis.techtarget.com/definition/class) to database tables and Java [data types](https://searchmicroservices.techtarget.com/definition/data-type) to SQL data types and provides querying and retrieval.

**Java**

Java is a general-purpose programming language that is class-based, object-oriented and designed to have as few implementation dependencies as possible. It is intended to let application developers write once, run anywhere (WORA),[17] meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.[18] Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but it has fewer low-level facilities than either of them. As of 2018, Java was one of the most popular programming languages in use according to GitHub particularly for client-server web applications, with a reported 9 million developers.

**System Analysis**

**System Analysis**

System analysis is a detailed study of various operations performed by a system and their relationships within and outside of the system. During analysis, data are collected on the available files, decisions points, and transactions handled by the present system.

**Why needed?**

System analysis is analyzing the system from different angles. The analysis of the system very necessary for an efficient system design. Through a proper system analysis the quality of system design can be modified and improved.

**System Analyst:**

System Analyst analyzes how well a software or hardware fit into the business needs of the clients. Computer System Analysts study an organization’s current computer systems, procedures and design information systems solution to help the organization work more efficiently and effectively. An analyst performs analysis and design techniques to find out if the instruction needs are met.

**Basic Task of System Analyst**

Basic tasks of system analyst are:

* Defining requirement
* Prioritizing requirement by consensus
* Analysis and evaluation
* Solving problems
* Drawing up specifications
* Designing systems
* Evaluating
* systems

**Procedural Steps of System Analysis**

There are three steps of analysis. They are:-

* Analysis Strategy: Analysis strategy defines the way works on making the system effective is done.
* Requirements Collecting: Requirements collecting in several ways. Requirements are building blocks for analysis.
* Proposal Presentation: All the steps make the analyst able to present a proposal.

**Techniques and Tools for Requirements Collecting**

Requirements collections requires various efforts and different techniques. Some of the strategies are:-

* Interviewing the client
* Perform survey
* Previous and current document analysis
* Observing client
* Questionnaires
* Brainstorming

**User of the system**

Users are of two kinds. They can be:-

* Authorized User:

Authorized users are those who can visit the page with their own log in and password. They can see what is in the Web pages, also they can perform change to the documents to the extent they are allowed to.

* Unauthorized Users:

Unauthorized users are those who don’t have any log in id, they only can view what is displayed in the web page and cannot perform any change.

**System Planning**

System planning consists of two parts-

* Feasibility Study
* Work Plan

**Feasibility Study**

It is a test of a system proposal according to its workability, impact on the organization, ability to meet user needs, and effective use of resources. This tests the visibility of the system. This check is performed throughout the life cycle of the system. Feasibility study can be of following types:

* Technical Feasibility: This is concerned with the technical fields, like input, output, process, procedures etc. This is good for long-term planning and trouble shooting.
* Economic Feasibility: This feasibility is concerned with the effectiveness of the system and the way it can be conducted to finish it within the budget.
* Operational Feasibility: Operational feasibility is the measurement of how well the system answers to the requirements. By this study we can say how much error-free the library package has been and how correctly the data has been calculated.
* Schedule Feasibility: Schedule feasibility concerns with the project work plan being able to be finished within the target time limit. The project need to finish within time, then it will support schedule feasibility.

**Work Plan**

Work plan is organization of the tasks to be performed for the project completion. The work plan needs to be within the time and budget. It has to be built such a way that it becomes responsive to the requirements and many survive in the long run.

**Adopted Analysis Strategy**

The existing system is very much time consuming and complicated to build. The strategy that is fast and simple. If the project is completed according to the plan then it can it can fulfill the expectation that cannot be achieved from the existing system. Therefore an absolute ending data can also be calculated.

**System Design**

The most creative and challenging phase. It describes a final system and the process by which it is developed. It refers to the technical specification that will be applied in implementation of the candidate system. The Enterprise Resource Planning System as a whole for the project is given in figure 1.

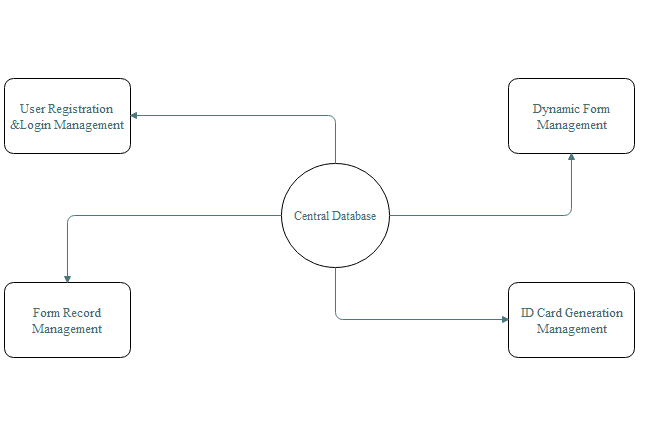


Fig. 6.1

**Design Necessity**

During design, customer requirements, business needs, and technical considerations all come together in the formulation of the system. Creates a model of the software by providing the detail about the software data structures, architectures, interfaces, and components that are necessary to implement the system. Each of the elements of the analysis model is translated to the corresponding design model to create the four design models required for a complete specification of design.

**Database Design**

Database is the basic repository of information about the actors. Database for our system is designed with SQLite. In the pattern used here, MVT: - Model, View and Template, Database are known as modeled. These models are built with SQLite which is the default database management system of intelliJ IDEA.

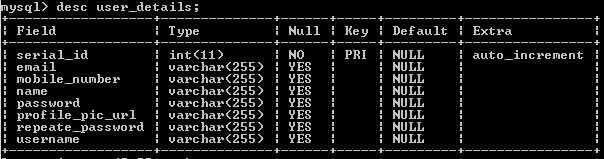
**Database Design Process**

Databases are designed following some steps:

* Database names are defined with the attributes and relations with other databases in the model area.
* Database tables are filled with data entries.

**Database Tables**

Database tables are mainly the data repositories containing the data. An example database table of our project is:



**Object Oriented Designs**

For constructing the system and structural development of the system Object Oriented Designs are done. This design process gives us the ability to view the system from different perspective. Different diagrams show the Object Oriented Designs. Among them:

* Context Diagram: Shows the relationship that the system has with other external entities
* Use case Diagram: Shows the functionalities.
* Sequence Diagram: Shows the work flow.
* Class Diagram: Shows the class sequence.

**Context Diagram**

The context diagram shows the system under consideration as a single high-level process and then shows the relationship that the system has with other external entities such as systems, organizational groups, external data stores etc. Another name for a Context Diagram is a Context-Level Data-flow Diagram or a Level-0 Data Flow Diagram. Since a Context Diagram is a specialized version of Data-Flow Diagrams can be helpful.

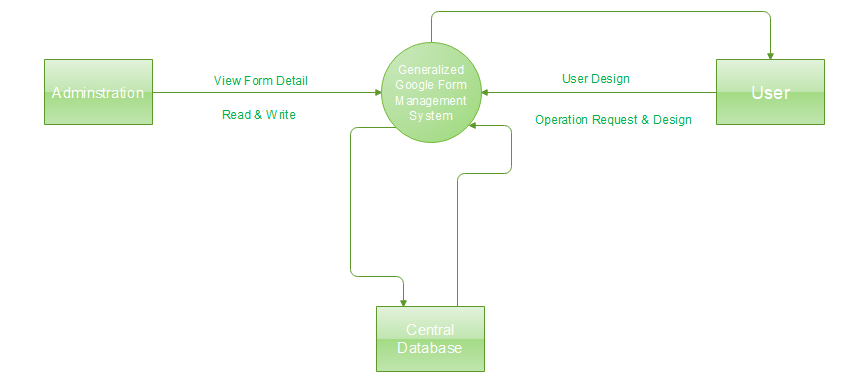


Fig. num.num Context Diagram of the System

**Use case diagram**

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.

They also help identify any internal or external factors that may influence the system and should be taken into consideration. They provide a good high level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented

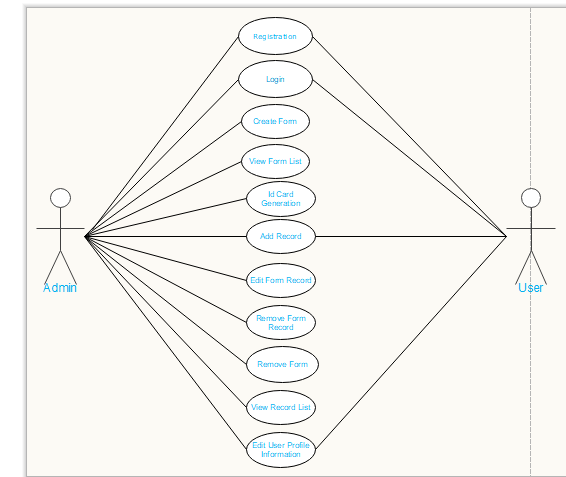


Fig. num.num Use case Diagram of the system.

**Class diagram**

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. This modeling method can run with almost all Object-Oriented Methods. A class can refer to another class. A class can have its objects or may inherit from other classes.

UML Class Diagram gives an overview of a software system by displaying classes, attributes, operations, and their relationships. This Diagram includes the class name, attributes, and operation in separate designated compartments. Class Diagram helps construct the code for the software application development.

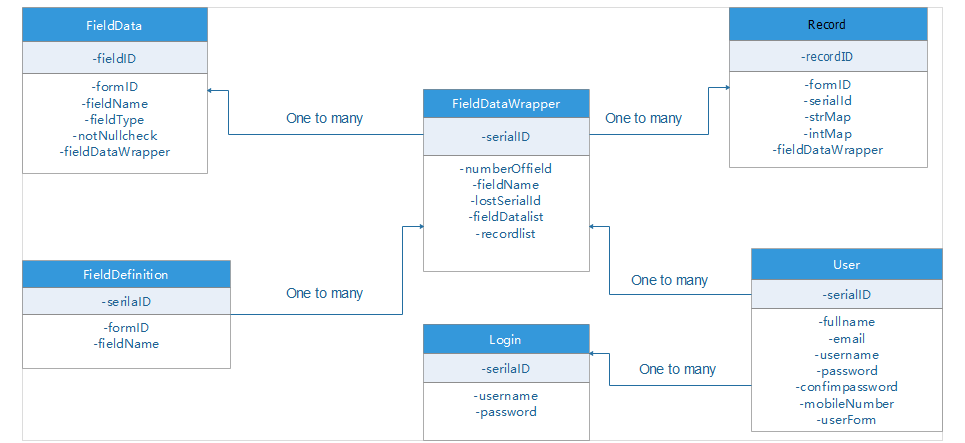


Fig. num.num Class diagram of the system.

**Sequence diagram**

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis.

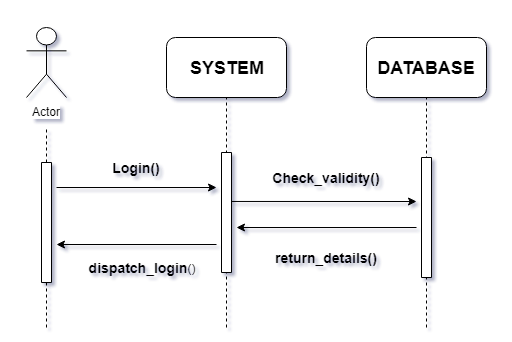
****

Fig. Sequence Diagram for Login

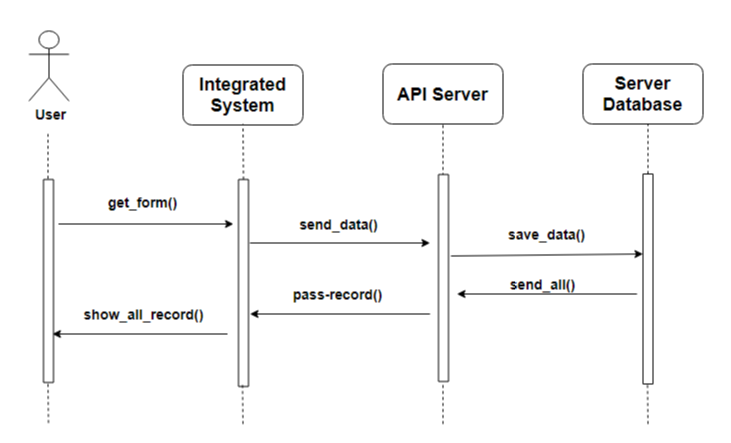


Fig. Sequence Diagram for Record Saving

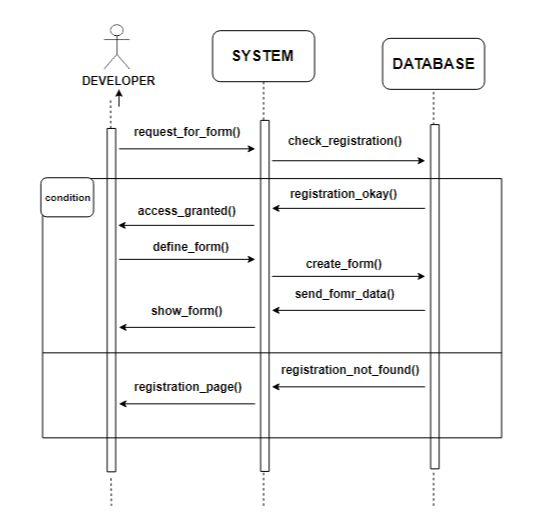


Fig. Sequence Diagram for Form Creation

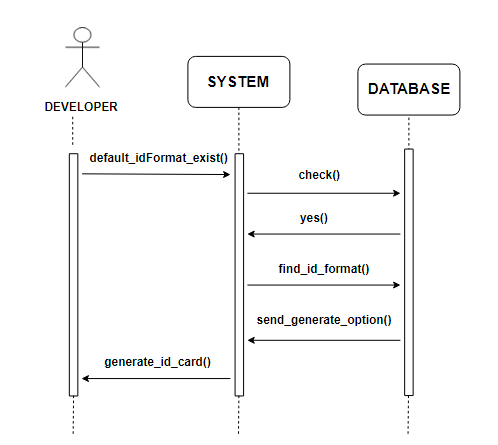


Fig. Sequence Diagram for ID card Generation

**User Interface Design**

User interface is one of the most important part of the system. While designing this various aspects are needed to be kept in mind. Like- the users are the novices, they need to get a friendly interface to operate the system. So, the interface needs to be simple.

**Design Flexibility**

The design of the user interface need to be flexible. So that the users can understand this and use the system fast. The design of our interface is user friendly and flexible to operate. Even any gust user can understand the system functions seeing the interface.

**Programming Pattern**

For programming or, say, coding MVT pattern is used. It is quite new for our use. MVT has similarities with MVC which helps a lot.

**MVT pattern**

MVT stands for Model, View, and Template. Model here depicts the database.

* Model (M): In the model part the databases are created with different attributes and different relationships with other databases.
* View (V): The view part is the main coding part. View part helps us to organize the things the way we want to see the components organized.
* Template (T): Template is the design part. It controls the user interface design and different pages design for view. It is done with HTML, CSS and Bootstrap, Thyme leaf.



Fig. num.num Component Interaction within MVT Pattern

**Component Interaction Regarding This Project**

Databases or, in this case, models are given to the system by the developer. The view and the template then just maps the models to some urls and spring serve the magic to the user. The component interaction within the MVT pattern can be showed with figure

**System Development**

This chapter consists of the brief description of how the system is developed under which environment, the Stages of the system development etc. These are all important topics with respect to system construction.

**Why perform System Development?**

System development is the process of defining, designing, testing and implementing a new system. It could include the internal development of customized systems, the creation of database systems or the acquisition of third party developed system.

**System Environment**

System environment is primarily the set of variables that define or control certain aspects of process execution. They are set or reset each time a shell is started. The system environment is the term commonly uses to refer to support an application. A system environment for a particular application could include:

\*Operating system: Windows 10.

\*Database system: SQLite.

\*Development Tools or Compiler: Bootstrap, CSS, HTML.

**Development Strategies**

There are various development approaches, like-

* Structured approach,
* Object oriented approach etc.

Our project is analyzed and preformed the development process in the object oriented approach. Object oriented approach views a system as collections of interacting objects that work together to accomplish a task. This interactions are known as use cases. Object-oriented development uses a UML class diagram to show all the classes of objects that are in the system.

It has two features that shows the usefulness of using the object-oriented approach, they are- naturalness and re-usability. Naturalness is that we usually think anything in the form of tangible objects. Re-usability is that this objects or classes can be used again and again.

**Software development life cycle**

Software development life cycle is a systematic approach to develop software. It creates a structure for the developer to design, create and deliver high quality software according to the requirements of customer or end user. It also provides a methodology for improving the quality of the desired product. The purpose of SDLC process is to provide help in producing a product that is cost efficient and of high quality.

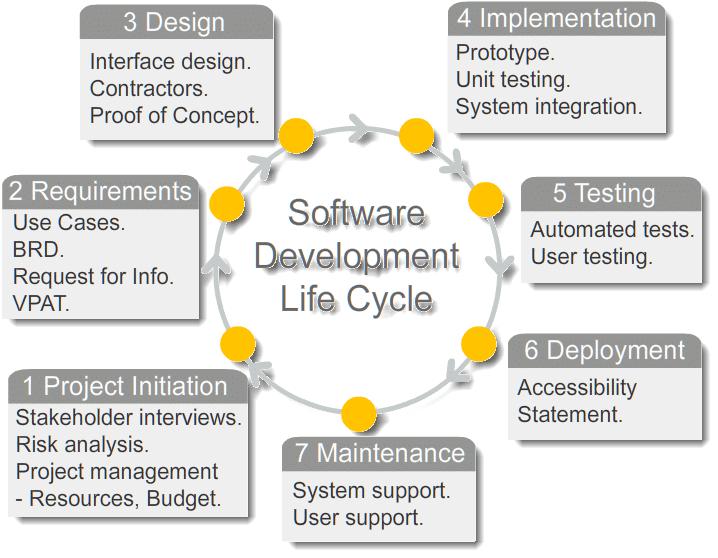


Fig. Software development life cycle.

**Stage 1. Planning:**

Planning is the requirements gathering phase. The requirements for our project has been gathered through questionnaires, brain storming etc.  Without the perfect plan, calculating the strengths and weaknesses of the project, development of software is meaningless. Planning kicks off a project flawlessly and affects its progress positively.

**Stage 2. Analysis:**

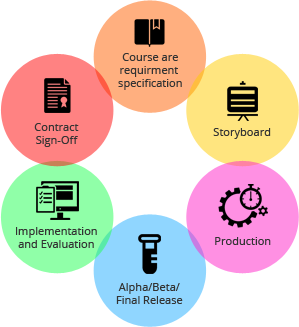
Analysis of our project has followed the object-oriented analysis approach. Drawing different UML diagrams a visualization has been made. This step is about analyzing the performance of the software at various stages and making notes on additional requirements. Analysis is very important to proceed further to the next stage.

**Stage 3. Design:**

Design of our project has followed also the object-oriented approach. Once the analysis is complete, the step of designing takes over, which is basically building the architecture of the project. This step helps remove possible flaws by setting a standard and attempting to stick to it.

**Stage 4. Development & Implementation:**

The actual task of developing the software starts here with data recording going on in the background. Once the software is developed, the stage of implementation comes in where the product goes through a pilot study to see if it’s functioning properly. The implementation of the project has been done with Java and spring boot.



**Stage 5: Testing**

This is the last phase of SDLC before the software is delivered to the customer. The job of test team is to test the system against the requirements. The aim of tester is to find out the gaps or defects within the system and also to verify that the software works as expected according to the requirements. It includes Unit testing, Integration testing and System testing.

**Stage 6. Maintenance**

Maintenance of the project is done through different perspectives. Maintenance like- corrective, adaptive, perfective. Once the software passes through all the stages without any issues, it is to undergo a maintenance process wherein it will be maintained and upgraded from time to time to adapt to changes. Almost every software development Indian company follows all the six steps, leading to the reputation that the country enjoys in the software market today.

**Implementation of System**

Implementation of the system has been done with MVT pattern, about which a description has been given in previous chapters. The framework is the essential supporting structures, in our project, it is Spring boot. A simple glimpse of the use of the spring boot is given in the following sections.

**Spring boot framework**

The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform.

Spring Boot provides a good platform for Java developers to develop a stand-alone and production-grade spring application that we can just run. We can get started with minimum configurations without the need for an entire spring configuration setup.

**Goals of Spring Boot**

Spring Boot is designed with the following goals −

1. To avoid complex XML configuration in spring,
2. To develop a production ready spring applications in an easier way,
3. To reduce the development time and run the application independently,
4. Offer an easier way of getting started with the application.

**How does it work?**

Spring Boot automatically configures your application based on the dependencies you have added to the project by using @EnableAutoConfiguration annotation. For example, if MySQL database is on your class path, but you have not configured any database connection, then Spring Boot auto-configures an in-memory database.

The entry point of the spring boot application is the class contains @SpringBootApplication annotation and the main method.Spring Boot automatically scans all the components included in the project by using @ComponentScan annotation.

**Spring Boot Starters**

Handling dependency management is a difficult task for big projects. Spring Boot resolves this problem by providing a set of dependencies for developer’s convenience.

For example, if we want to use spring and JPA for database access, it is sufficient if we include spring-boot-starter-data-jpa dependency in our project. Note that all Spring Boot starters follow the same naming pattern spring-boot-starter- \*, where \* indicates that it is a type of the application.

**Example**

Look at the following Spring Boot starters explained below for a better understanding. Spring Boot Starter Actuator dependency is used to monitor and manage your application. Its code is shown below −

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

Spring Boot Starter Security dependency is used for Spring Security. Its code is shown below −

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-security</artifactId>

</dependency>

Spring Boot Starter web dependency is used to write a Rest Endpoints. Its code is shown below −

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

Spring Boot Starter Thyme Leaf dependency is used to create a web application. Its code is shown below −

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

Spring Boot Starter Test dependency is used for writing Test cases. Its code is shown below −

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test<artifactId>

</dependency>

Source: (https://www.tutorialspoint.com/spring\_boot/spring\_boot\_introduction)

**Advantages of Spring Boot:**

1. It is very easy to develop Spring Based applications with Java or Groovy.
2. It reduces a lot of development time and increases productivity.
3. It avoids writing of boilerplate Code, Annotations and XML Configuration.
4. It is very easy to integrate Spring Boot Application with its Spring Ecosystem like Spring JDBC, Spring ORM, Spring Data, Spring Security etc.
5. It follows “Opinionated Defaults Configuration” Approach to reduce Developer effort
6. It provides Embedded HTTP servers like Tomcat, Jetty etc. to develop and test our web applications very easily.
7. It provides CLI (Command Line Interface) tool to develop and test Spring Boot (Java or Groovy) Applications from command prompt very easily and quickly.
8. It provides lots of plugins to develop and test Spring Boot Applications and it uses Build Tools like Maven and Gradle.

Source of advantages: (http://www.pro-tekconsulting.com/blog/what-are-the-advantages using-spring-boot).

**Testing and Maintenance of the System**

This chapter encloses the after works of system construction. It gives a brief view of the testing of the system and the maintenance strategy. It is a very important State for the success of the system.

**System Testing**

System Testing is testing the components of the system and the overall testing of the system. It comprises of various types of testing. The figure below shows the testing strategy required for a system testing:

**System Testing Activities**

System Testing activities include:

1. Unit Testing: Performed on unit or simple components of code.

2. Integration Testing: Performed on a module, consisting of different units.

3. Validation Testing: Performed to identify if the requirements are fulfilled or not.

4. System Testing: it is the overall testing to check if the system is working or not.

**Necessity of System Testing:**

It is necessary to assure the correct functioning of the system, so system testing is necessary. To ensure that the system is built with less complexity and if there is any fault or error that can be recovered in this stage. Testing is a vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. System testing is its utility as a user-oriented vehicle before implementation. The best Program is worthless if it does not meet user needs.

**Maintenance**

Maintaining the system is an important aspect. As key personal change positions in the Organization, new changes will be implemented. The need for maintenance is severe when it comes the need for future update according to the need of future situation.

**Different Types of Maintenance**

There are different types of maintenance for any system. They are:

1. Corrective Maintenance.
2. Adaptive Maintenance.
3. Perfective Maintenance.

**Corrective Maintenance**

It is correction of software faults and failures. In case of our project it will be:

\* Correcting the Google form if any change found from the collection of updated information.

\* Make the Corrections if any case becomes wrong in different circumstances.

\* Control database redundancy as database manipulators are many.

**Adaptive Maintenance**

Adapting the current system to additional circumstances without changing the system is adaptive maintenance. In case of our project adaptive maintenance will be:

\* In case of redundant information, managing huge number of students, employees, staff and multiple sectors.

\*Reorganizing the Google form, if unwanted changes come up.

**Perfective Maintenance**

Enhancement and improvement of the existing system with respect to locally limited issues. This will answer the further requirements of the users. This maintenance answers to the outside or environmental changes.

**Necessity of Maintenance**

Maintenance has a vital role in system construction and upgrading. The system requires regular update, which maintenance does. In case of any failure because of any faults conducted by the users, maintenance is required. To handle a vast amount of users then the regular user amount maintenance is required.

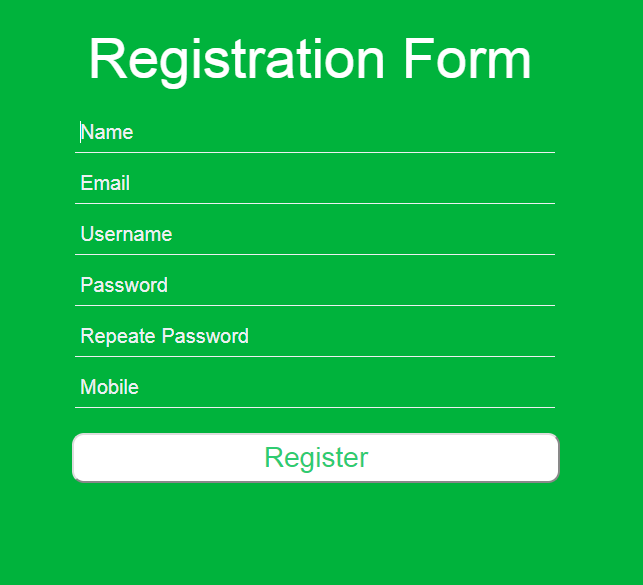
In this section, design of data of the system is performed. It is important part to maintain the system. The entity Relationship diagram is a summary of the various data objects that make up the web based library system. Included in each table are the attributes of each object, the data type for each attribute, the number of characters allowed for each field, the default value and any other information that defines the fields.

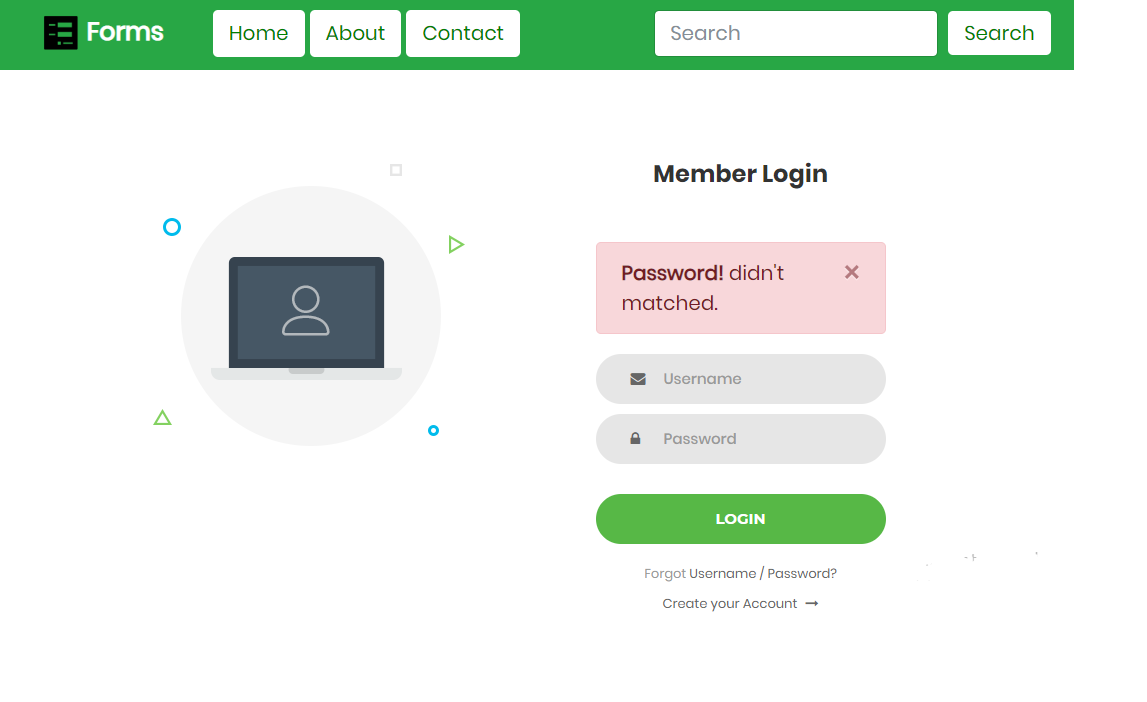
**Result & Discussion**

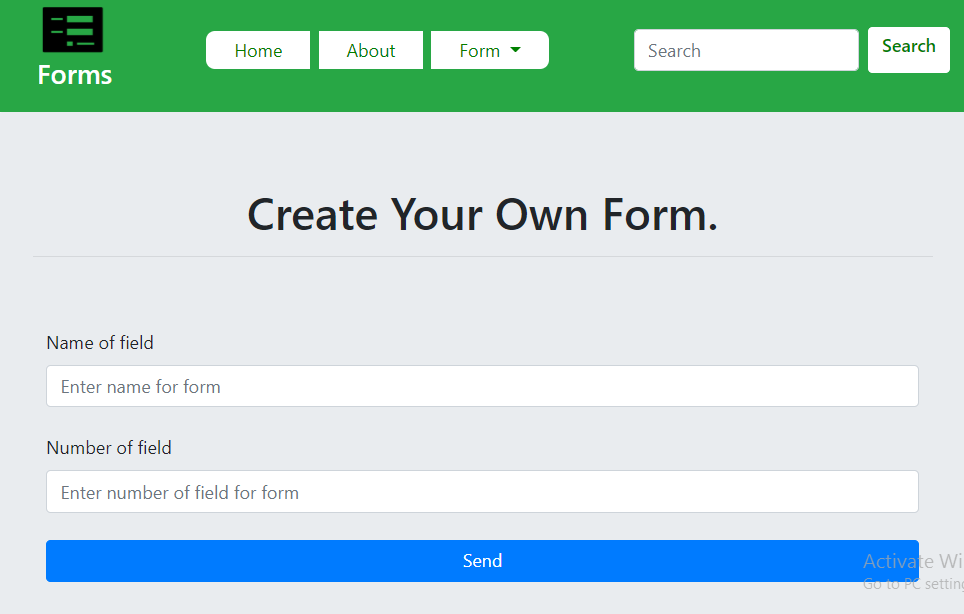
In this chapter we show the output as result and a simple discussion about the result and its uses. Describes a discussion for the project.

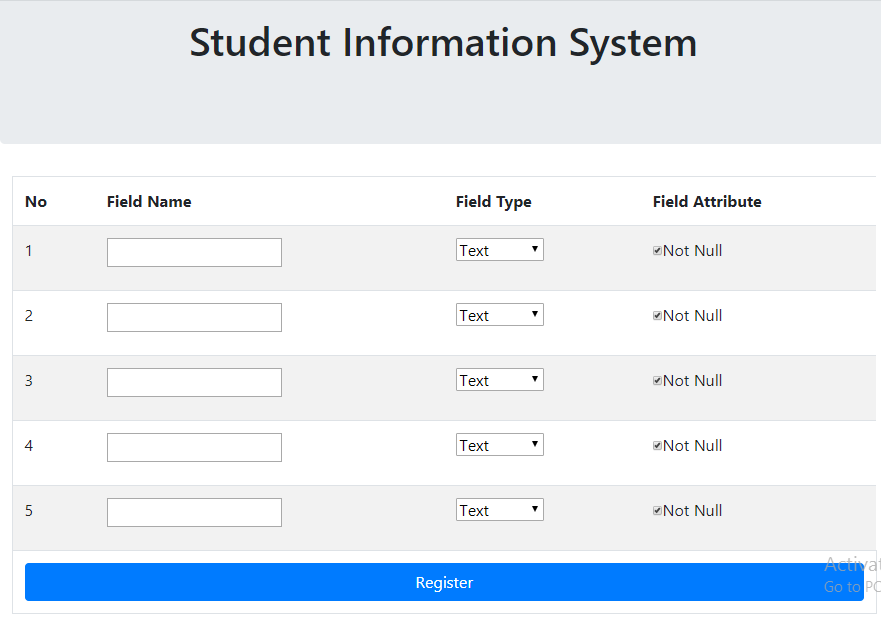
**Result**

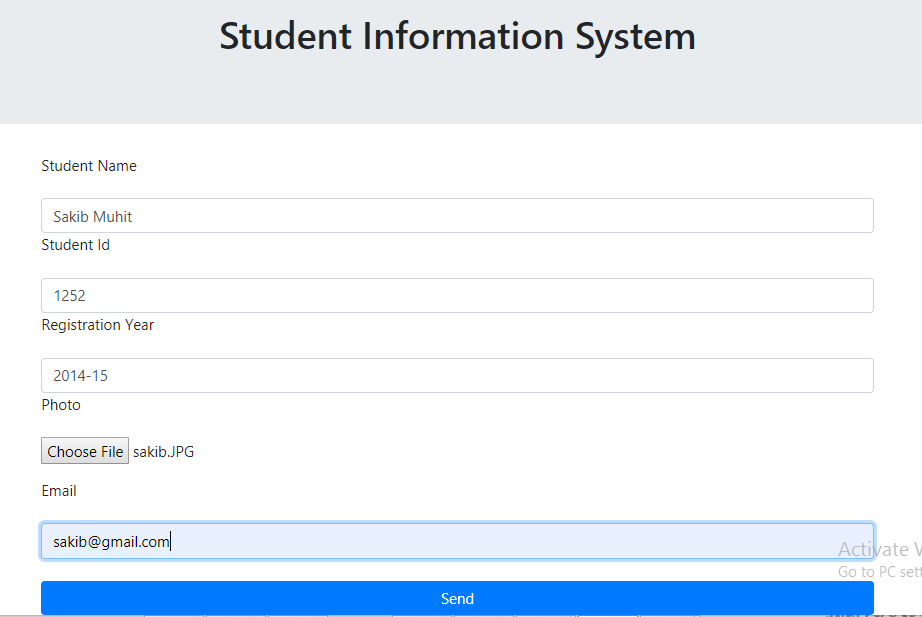
After the compilation of the project we will get API as the output. Let’s have a view of the project.

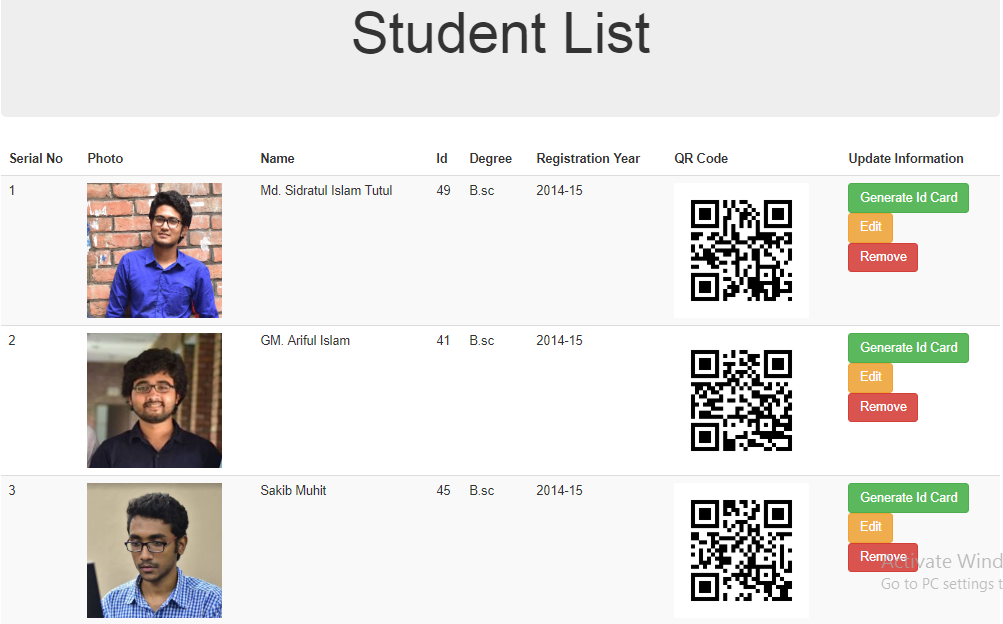


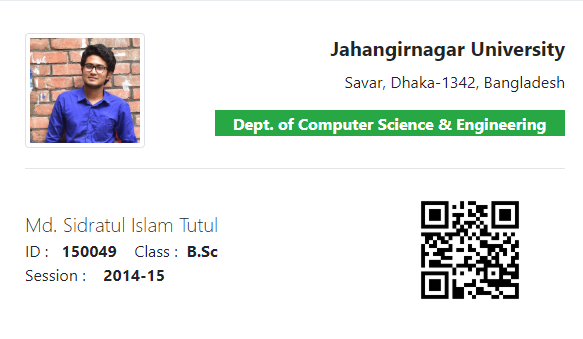












**Discussion**

This project has the quality factors ensuring it’s quality. It is expandable for large number of users. It has flexibility in design as virtual environment is created for building the system, the system can run on different environment.

**Conclusion and Future Work**

All the details of the data has been kept in the database. The project implementation in real life and future plan regarding the project has been covered in this chapter. This is the final chapter closing the different aspects of the project.

**Project Benefits**

This project is built for multipurpose. The success of the project effects an educational institute in a good way. The benefits of the project can be viewed as:

* User details come within a computerized format, which is helpful, fast and less tiring then the existing system.
* The lengthy process of making a form for storing information becomes short and visibly faster.
* The system, being responsive, increases the commitment of the developers to the system. And regulates the work process in the usual flow.

**Future Work**

The future work for this project involves a great advancement in the technology. Certain features will be added and custom interface will be given for developers to generate ID card for any information system.

**Personal Achievements**

While building the system multiple achievements enriched us. It has increased the list of our personal achievements. Among them:

* Learned a new framework Spring Boot, Thyme leaf.
* Come to know an IDE intelliJ IDEA.
* Learn to use the design tools HTML, CSS, Bootstrap in a new field.
* A new coding pattern MVT has also known.
* Multiple organizations of databases is learned.

**Conclusion**

At the end of the project we can say that the construction of the project has enlightened us with the new area of developing. Make us known with new patterns and framework. The project has a great impact for information management. It increases the working capacity, by making the form faster and in a less complicated way.