**Policy Management System**

**A**

**Project Report**

**Submitted in Complete Fulfillment of the Requirement for the Award of Degree**

**BACHELOR OF TECHNOLOGY**

**IN**

**Computer Science and Technology**

Submitted By:

**NAME WBUT ROLL NO:**

**Arya Dasgupta 14800115024**

**Under the Guidance of**

**(Mentor, Cognizant Academy)**

**Krishnendu Ganguly (Manager, Cognizant Academy)**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**FUTURE INSTITUTE OF ENGINEERING AND MANAGEMENT**

**KOLKATA-700150, INDIA**

**Table Of Contents**

1. Acknowledgement
2. Aim Of the Project
3. Project Overview
4. Methodology to be followed

* Agile Software Development Principles

Features:

* Iterative,Incremental and evolutionary
* Efficient and Face to face communications
* Very Short feedback loop and adaptation cycle
* Quality focus

1. Detail Project Description

* Vendor Registration and Vendor Login
* Customer Registration and Customer Login

1. Technology Used
2. Conclusion
3. Bibliography

**ACKNOWLEDGEMENT**

It is a great pleasure for us to express our earnest and great appreciation to **Debasri Mukherjee**(our mentor) , **Rahul Pramanik** (Batch Owner at Cognizant Academy) , **Krishnendu Ganguly** (Manager of Cognizant Academy) . We are very much grateful **to Debasri Mukherjee** for her kind guidance, encouragement, valuable suggestions, innovative ideas, and supervision throughout this project work, without which the completion of the project work would have been difficult one and **Krishnendu Ganguly** for giving us this fantastic opportunity .

We would like to express our thanks to the Head of the Department, Prof. **Jagannibas Paul Choudhury** for his active support.

Also we would like to sincerely appreciate:

1. **Debasri Mukherjee**

for all the efforts she has put in during the early phase of our internship that helped us a lot throughout the entire internship program.

We also express our sincere thanks to all the teachers of the department for their precious help, encouragement, kind cooperation and suggestions throughout the development of the project work.

We would like to express our gratitude to everyone at Cognizant GTP for providing us with a congenial working environment.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(**Debasri Mukherjee** – Mentor) (**Krishnendu Ganguly**-Manager Academy)

**AIM OF THE PROJECT**

This document is aimed at:

• Providing the necessary inputs to the detailed requirements gathering phase and further on for the SDLC processes.

• This document also serves to establish the traceability between the Business Objectives and the requirements identified in the proposed solution and how they satisfy the stated objectives.

• Provide expectation traceability in terms of the requirements and the user expectation

• Serves as a formal template for documenting the Business Requirements which also includes statutory and regulatory requirements.

The purpose of this document is to systematically capture requirements for the project and the system to be developed. Functional requirements are captured in this document. It also serves as the input for the project scoping

# PROJECT OVERVIEW

Below are the objectives that shall be fulfilled post the execution of this project:

Policy Management system will create and maintain Customer and Vendor information.

* Create and maintain customer and vendor information.
* The vendor can add new policies, edit the policies and delete the policies.
* Customer can search for policies.
* Customer can buy new policies and remove the policies he already had.

Whenever vendor registers, their credentials are to be stored in a database with an auto generated Vendor Id. Whenever customer registers, their credentials are stored in the database. While registering certain validations like does the username already exists is checked and if it does exist then the customer/vendor will not be allowed to register. Also, they can login if they are already registered.

The Vendor will be able to add new policies to the database, edit the existing policies and also remove the policies from the database.

The Customers can search for policies uploaded by the vendor in the database. He can buy new policies and also remove the policies he possesses.

This application uses HTML5, CSS3, Bootstrap4, JSP on the front end and Spring MVC in the middleend and Hibernate to establish database connectivity. Here we made use of Oracle 11g.

# METHODOLOGY TO BE FOLLOWED

In this project we are going to follow **Agile** methodology

AGILE SOFTWARE DEVELOPMENT is an approach to [software development](https://en.wikipedia.org/wiki/Software_development) under which

requirements and solutions evolve through the collaborative effort of [self-organizing](https://en.wikipedia.org/wiki/Self-organization#Human_society) andcross functionalteams and their [customer(s)](https://en.wikipedia.org/wiki/Customer)[/end user(s).](https://en.wikipedia.org/wiki/End_user) it advocates adaptive planning, evolutionary development, [empirical knowledge,](https://en.wikiquote.org/wiki/Empirical_knowledge) and [continual improvement,](https://en.wikipedia.org/wiki/Continual_improvement_process) and it encourages rapid and flexible response to change.

The term agile (sometimes written agile) was popularized, in this context, by the [manifesto for agile software development.](https://en.wikipedia.org/wiki/Agile_software_development#The_Agile_Manifesto) the values and principles espoused in this manifesto were derived from and underpin a broad range of [software development frameworks,](https://en.wikipedia.org/wiki/Software_development_process) including [scrum](https://en.wikipedia.org/wiki/Scrum_(software_development)) and KANBAN [.](https://en.wikipedia.org/wiki/Kanban_(development))

There is significant anecdotal evidence that adopting agile practices and values improves the agility of software professionals, teams and organizations; however, some empirical studies have found no scientific evidence.

# AGILE SOFTWARE DEVELOPMENT PRINCIPLES

The manifesto for agile software development is based on twelve principles:

1. Customer satisfaction by early and continuous delivery of valuable software.
2. Welcome changing requirements, even in late development.
3. Deliver working software frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

**FEATURES OF AGILE PRINCIPLES:**

# ITERATIVE, INCREMENTAL AND EVOLUTIONARY

Most agile development methods break product development work into small increments that minimize the amount of up-front planning and design. iterations, or sprints, are short time frames [(time boxes)](https://en.wikipedia.org/wiki/Timeboxing) that typically last from one to four weeks. each iteration involves a [cross-functional team](https://en.wikipedia.org/wiki/Cross-functional_team) working in all functions: [planning,](https://en.wikipedia.org/wiki/Project_planning) [analysis,](https://en.wikipedia.org/wiki/Requirements_analysis) [design,](https://en.wikipedia.org/wiki/Software_design) [coding,](https://en.wikipedia.org/wiki/Computer_programming) [unit testing,](https://en.wikipedia.org/wiki/Unit_testing) and [acceptance testing.](https://en.wikipedia.org/wiki/Acceptance_testing) at the end of the iteration a working product is demonstrated to stakeholders. this minimizes overall risk and allows the product to adapt to changes quickly. an iteration might not add enough functionality to warrant a market release, but the goal is to have an available release (with minimal [bugs)](https://en.wikipedia.org/wiki/Software_bug) at the end of each iteration. multiple iterations might be required to release a product or new features. working software is the primary measure of progress.

# EFFICIENT AND FACE-TO-FACE COMMUNICATION

The principle of [co-location](https://en.wikipedia.org/wiki/Colocation_(business)) is that co-workers on the same team should be situated together to better establish the identity as a team and to improve communication. this enables [face-to-face interaction,](https://en.wikipedia.org/wiki/Face-to-face_interaction) ideally in front of a whiteboard, that reduces the cycle time typically taken when questions and answers are mediated through phone, persistent chat, wiki, or email.

no matter which development method is followed, every team should include a [customer representative](https://en.wikipedia.org/wiki/Customer_representative) ("product owner" in [scrum)](https://en.wikipedia.org/wiki/Scrum_(software_development)). this person is agreed by stakeholders to act on their behalf and makes a personal commitment to being available for developers to answer questions throughout the iteration. at the end of each iteration, stakeholders and the customer representative review progress and re-evaluate priorities with a view to optimizing the [return on investment](https://en.wikipedia.org/wiki/Rate_of_return) (ROI) and ensuring alignment with customer needs and company goals.

in agile software development, an **information radiator** is a (normally large) physical display located prominently near the development team, where passers-by can see it. it presents an up-to date summary of the product development status. a [build light indicator](https://en.wikipedia.org/wiki/Build_light_indicator) may also be used to inform a team about the current status of their product development.

# VERY SHORT FEEDBACK LOOP AND ADAPTATION CYCLE

A common characteristic in agile software development is the [daily stand-up](https://en.wikipedia.org/wiki/Stand-up_meeting) (also known as the daily scrum). in a brief session, team members report to each other what they did the previous day toward their team's iteration goal, what they intend to do today toward the goal, and any roadblocks or impediments they can see to the goal.

# QUALITY FOCUS

Specific tools and techniques, such as [continuous integration,](https://en.wikipedia.org/wiki/Continuous_integration) automated [unit testing,](https://en.wikipedia.org/wiki/Unit_testing) [pair programming,](https://en.wikipedia.org/wiki/Pair_programming) [test-driven development,](https://en.wikipedia.org/wiki/Test-driven_development) [design patterns,](https://en.wikipedia.org/wiki/Software_design_pattern) [behavior-driven development,](https://en.wikipedia.org/wiki/Behavior-driven_development) [domain](https://en.wikipedia.org/wiki/Domain-driven_design) [driven design,](https://en.wikipedia.org/wiki/Domain-driven_design) [code refactoring](https://en.wikipedia.org/wiki/Code_refactoring) and other techniques are often used to improve quality and enhance product development agility. this is predicated on designing and building quality in from the beginning and being able to demonstrate software for customers at any point, or at least at the end of every iteration.

# DETAIL PROJECT DESCRIPTION

The full project is divided into two segments. They are as follows:

* Vendor Registration and Vendor Login
* Customer Registration and Customer Login

# Vendor Registration and Vendor Login

We have a designed a separate webpage that opens when the user selects himself as a vendor.

The vendor can login using his credentials if he has already registered himself. While registering the vendor will be provided with an auto generated id which the vendor uses to login in the future.

Vendor functionalities involve the following:

* Add new policies.
* Edit the existing policies.
* Remove the policies.

Let’s discuss each functionality in details:

* **Add new policies**

The vendor will be able to add new policies into the database which can be viewed by the customer.

* **Edit the existing policies**

The vendor can edit the existing policies that has been already uploaded.

* **Remove the policies**

In this section the vendor can remove the policies which are already present in the database.

# Customer Registration and Customer Login

This part gives access to the Customers. The Customer can login using his credentials if he has already registered himself. If the email-id already exists then the Customer will not be allowed to register. The email-id and the password provided by the user during registration will be used by the customer for future login.

The various functionalities involve:

* + Search for policies.
  + Buy new policies.
  + Remove the policies that he already possesses.
  + View policies.

Let’s discuss the functionalities all one by one:

* **Search for policies**

The customer has the ability to search for policies which has been already added by the vendor before adding new policies. All the policies that will be added by the vendors will be available for the customer to watch and select his required policies.

* **Buy new policies**

The customer after searching and going through the policies can select a policy to buy and can buy the policy.

* **Remove the policies that he already possesses**

The customer will be able to remove the policies that he already has.

* **View policies**

Customer will be able to view the policies that the possesses and after viewing his policies he can decide whether he wants to remove the policies or not.

**TECHNOLOGIES USED**

#  HTML5

**HTML 5** (formerly and commonly spelled **HTML5**) is a [software solution stack](https://en.wikipedia.org/wiki/Solution_stack) that defines the properties and behaviors of [web page](https://en.wikipedia.org/wiki/Web_page) [content](https://en.wikipedia.org/wiki/Web_content) by implementing a [markup](https://en.wikipedia.org/wiki/Markup_language) based [pattern](https://en.wikipedia.org/wiki/Software_design_pattern) to it. HTML 5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and [application programming interfaces](https://en.wikipedia.org/wiki/Application_programming_interface) (APIs) for complex [web applications](https://en.wikipedia.org/wiki/Web_application).[[11]](https://en.wikipedia.org/wiki/HTML5#cite_note-HTML5diffHTML4-12) For the same reasons, HTML 5 is also [a candidate for cross-platform mobile applications](https://en.wikipedia.org/wiki/HTML5_in_mobile_devices), because it includes features designed with low-powered devices in mind. Many new [syntactic](https://en.wikipedia.org/wiki/Syntax_(programming_languages)) features are included. To natively include and handle [multimedia](https://en.wikipedia.org/wiki/Multimedia) and [graphical](https://en.wikipedia.org/wiki/2D_computer_graphics) content.

**FEATURES**

HTML 5 introduces [elements](https://en.wikipedia.org/wiki/HTML_element) and attributes that reflect typical usage on modern [websites](https://en.wikipedia.org/wiki/Website). Some of them are semantic replacements for common uses of generic block (<div>) and inline (<span>) elements, for example <nav> (website navigation block), <footer> (usually referring to bottom of web page or to last lines of HTML code), or <audio> and <video> instead of <object>. Some deprecated elements from [HTML 4.01](https://en.wikipedia.org/wiki/HTML_4.01) have been dropped, including purely presentational elements such as <font> and <center>, whose effects have long been superseded by the more capable [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets). There is also a renewed emphasis on the importance of [DOM scripting](https://en.wikipedia.org/wiki/DOM_scripting) in Web behavior.

The HTML 5 syntax is no longer based on [SGML](https://en.wikipedia.org/wiki/Standard_Generalized_Markup_Language)despite the similarity of its markup. It has, however, been designed to be backward-compatible with common parsing of older versions of HTML. It comes with a new introductory line that looks like an SGML [document type declaration](https://en.wikipedia.org/wiki/Document_type_declaration), <!DOCTYPE html>, which triggers the standards-compliant [rendering mode](https://en.wikipedia.org/wiki/Quirks_mode).

#  JAVASCRIPT

**JavaScript**, often abbreviated as **JS**, is a [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) [programming language](https://en.wikipedia.org/wiki/Programming_language) that conforms to the [ECMAScript](https://en.wikipedia.org/wiki/ECMAScript) specification. JavaScript has [curly-bracket syntax](https://en.wikipedia.org/wiki/List_of_programming_languages_by_type#Curly-bracket_languages), [dynamic typing](https://en.wikipedia.org/wiki/Dynamic_programming_language), [prototype-based](https://en.wikipedia.org/wiki/Prototype-based_programming) [object-orientation](https://en.wikipedia.org/wiki/Object-oriented_programming), and [first-class functions](https://en.wikipedia.org/wiki/First-class_function).

Alongside [HTML](https://en.wikipedia.org/wiki/HTML) and [CSS](https://en.wikipedia.org/wiki/CSS), JavaScript is one of the core technologies of the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web). JavaScript enables interactive [web pages](https://en.wikipedia.org/wiki/Web_page) and is an essential part of [web applications](https://en.wikipedia.org/wiki/Web_application). The vast majority of [websites](https://en.wikipedia.org/wiki/Website) use it, and major [web browsers](https://en.wikipedia.org/wiki/Web_browser) have a dedicated [JavaScript engine](https://en.wikipedia.org/wiki/JavaScript_engine) to execute it.

As a multi-paradigm language, JavaScript supports [event-driven](https://en.wikipedia.org/wiki/Event-driven_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming), and [imperative](https://en.wikipedia.org/wiki/Imperative_programming) (including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [prototype-based](https://en.wikipedia.org/wiki/Prototype-based_programming)) [programming styles](https://en.wikipedia.org/wiki/Programming_paradigm). It has [APIs](https://en.wikipedia.org/wiki/Application_programming_interface) for working with text, [arrays](https://en.wikipedia.org/wiki/Array_data_type), dates, [regular expressions](https://en.wikipedia.org/wiki/Regular_expression), and the [DOM](https://en.wikipedia.org/wiki/Document_Object_Model), but the language itself does not include any [I/O](https://en.wikipedia.org/wiki/Input/output), such as [networking](https://en.wikipedia.org/wiki/Computer_network), [storage](https://en.wikipedia.org/wiki/Data_storage), or [graphics](https://en.wikipedia.org/wiki/Computer_graphics) facilities. It relies upon the host environment in which it is embedded to provide these features.

Initially only implemented [client-side](https://en.wikipedia.org/wiki/Client-side) in web browsers, JavaScript engines are now embedded in many other types of host software, including [server-side](https://en.wikipedia.org/wiki/Server-side) in web servers and databases, and in non-web programs such as word processors and [PDF](https://en.wikipedia.org/wiki/Portable_Document_Format) software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

Although there are similarities between JavaScript and [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), including language name, [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)), and respective [standard libraries](https://en.wikipedia.org/wiki/Standard_library), the two languages are distinct and differ greatly in design. JavaScript was influenced by programming languages such as [Self](https://en.wikipedia.org/wiki/Self_(programming_language)) and [Scheme](https://en.wikipedia.org/wiki/Scheme_(programming_language)).

* **SPRING MVC**
* A Spring MVC is a Java framework which is used to build web applications. It follows the Model-View-Controller design pattern. It implements all the basic features of a core spring framework like Inversion of Control, Dependency Injection.
* A Spring MVC provides an elegant solution to use MVC in spring framework by the help of **DispatcherServlet**. Here, **DispatcherServlet** is a class that receives the incoming request and maps it to the right resource such as controllers, models, and views.

**Advantages of Spring MVC Framework:**

Let's see some of the advantages of Spring MVC Framework:-

* **Separate roles** - The Spring MVC separates each role, where the model object, controller, command object, view resolver, DispatcherServlet, validator, etc. can be fulfilled by a specialized object.
* **Light-weight** - It uses light-weight servlet container to develop and deploy your application.
* **Powerful Configuration** - It provides a robust configuration for both framework and application classes that includes easy referencing across contexts, such as from web controllers to business objects and validators.
* **Rapid development** - The Spring MVC facilitates fast and parallel development.
* **Reusable business code** - Instead of creating new objects, it allows us to use the existing business objects.
* **Easy to test** - In Spring, generally we create JavaBeans classes that enable you to inject test data using the setter methods.
* **Flexible Mapping** - It provides the specific annotations that easily redirect the page.

## **Spring Web Model-View-Controller:**

## Spring MVC Tutorial

* **Model** - A model contains the data of the application. A data can be a single object or a collection of objects.
* **Controller** - A controller contains the business logic of an application. Here, the @Controller annotation is used to mark the class as the controller.
* **View** - A view represents the provided information in a particular format. Generally, JSP+JSTL is used to create a view page. Although spring also supports other view technologies such as Apache Velocity, Thymeleaf and FreeMarker.
* **Front Controller** - In Spring Web MVC, the DispatcherServlet class works as the front controller. It is responsible to manage the flow of the Spring MVC application.
* **JAVA SERVER PAGES**

**JavaServer Pages (JSP)** is a technology for developing Webpages that supports dynamic content. This helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.

A JavaServer Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through Webpage forms, present records from a database or another source, and create Webpages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

* **ORACLE**

An **Oracle** **database** is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a [**server**](https://docs.oracle.com/cd/B19306_01/server.102/b14220/glossary.htm#i432724) reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.  
Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.  
The database has **logical structures** and **physical structures**. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

* **CSS**

**Cascading Style Sheets**, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.  
CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

* **BOOTSTRAP**

**Bootstrap** is a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source) [CSS framework](https://en.wikipedia.org/wiki/CSS_framework) directed at responsive, mobile-first [front-end web development](https://en.wikipedia.org/wiki/Front-end_web_development). It contains [CSS](https://en.wikipedia.org/wiki/CSS)- and (optionally) [JavaScript](https://en.wikipedia.org/wiki/JavaScript)-based design templates for [typography](https://en.wikipedia.org/wiki/Web_design#Typography), [forms](https://en.wikipedia.org/wiki/Form_(HTML)), [buttons](https://en.wikipedia.org/wiki/Button_(computing)#HTML), [navigation](https://en.wikipedia.org/wiki/Web_navigation#Local_website_navigation) and other interface components.

Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to [web apps](https://en.wikipedia.org/wiki/Web_Apps)). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all [HTML elements](https://en.wikipedia.org/wiki/HTML_element). The end result is a uniform appearance for prose, tables and form elements across [web browsers](https://en.wikipedia.org/wiki/Web_browser). In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Bootstrap also comes with several JavaScript components in the form of [jQuery](https://en.wikipedia.org/wiki/JQuery) plugins. They provide additional user interface elements such as [dialog boxes](https://en.wikipedia.org/wiki/Dialog_box), [tooltips](https://en.wikipedia.org/wiki/Tooltip), and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields.

* **Hibernate**

**Hibernate ORM** (Hibernate in short) is an [object-relational mapping](https://en.wikipedia.org/wiki/Object-relational_mapping) tool for the [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) programming language. It provides a [framework](https://en.wikipedia.org/wiki/Software_framework) for mapping an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) domain model to a [relational database](https://en.wikipedia.org/wiki/Relational_database). Hibernate handles [object-relational impedance mismatch](https://en.wikipedia.org/wiki/Object-relational_impedance_mismatch) problems by replacing direct, [persistent](https://en.wikipedia.org/wiki/Persistence_(computer_science)) database accesses with high-level object handling functions.

Hibernate is [free software](https://en.wikipedia.org/wiki/Free_software) that is distributed under the [GNU Lesser General Public License](https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License) 2.1.

Hibernate's primary feature is mapping from Java classes to [database tables](https://en.wikipedia.org/wiki/Table_(database)), and mapping from Java data types to [SQL](https://en.wikipedia.org/wiki/SQL) data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set.

Hibernate can be used both in standalone [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) applications and in [Java EE](https://en.wikipedia.org/wiki/Java_EE) applications using [servlets](https://en.wikipedia.org/wiki/Java_Servlet), [EJB](https://en.wikipedia.org/wiki/Enterprise_JavaBeans) session beans, and [JBI](https://en.wikipedia.org/wiki/Java_Business_Integration) service components. It can also be included as a feature in other programming languages. For example, [Adobe](https://en.wikipedia.org/wiki/Adobe_Systems) integrated Hibernate into version 9 of [ColdFusion](https://en.wikipedia.org/wiki/ColdFusion) (which runs on J2EE app servers) with an abstraction layer of new functions and syntax added into [CFML](https://en.wikipedia.org/wiki/CFML).

* **Advantages of Hibernate**

1.Hibernate is database independent. You can work with any database you want like oracle,mysql,db2,sql server ,etc. Using hibernate you won't worry about writing database specific queries and syntax. It's providesHQL (Hibernate Query Language), which is compatible with any database server. You just need to write queries in HQL, at the endhibernate converts HQL to underlying database and executes i**t**.   
   
2. In ORM, you will map a database table with java object called "Entity". So once you map these,you will get advantages of OOP concepts like inheritance, encapsulation,etc.

3.Supports Lazy loading (also called n+1 problem in Hibernate). Take an example-parent class has n number of child class. So When you want information from only 1 child class,there is no meaning of loading n child classes.This is called lazy loading (Load only thing which you want).

**TECHNICAL REQUIREMENTS**

The following are the technical requirements for the project:

**Front-End**

* HTML5
* CSS
* Bootstrap
* JavaScript

**Middle-End**

* Spring MVC
* Hibernate

**Back-End**

* Oracle

**IDE Used:**

* Eclipse(Oxygen)

# CONCLUSION

Front-end ecosystem is constantly evolving and changing on a day-to-day basis. some tools become “bestsellers” in terms of web app development, revolutionizing the workflow, while others become a dead end.

HTML5, CSS and BOOTSTRAP are labelled as the best-seller, these have quickly won popular affection. Today it is maintained by the developer community and is used by all leading companies.

Its advantages can be listed as:

* It facilitates the overall process of writing components
* It boosts productivity and facilitates further maintenance
* It ensures faster rendering
* It is SEO friendly

# BIBLIOGRAPHY

We have received immense help from:

* <https://www.w3schools.com/js/default.asp>
* <https://www.w3schools.com/html/default.asp>
* <https://materializecss.com/about.html>
* <https://www.w3schools.com/jquery/default.asp>
* <https://www.tutorialspoint.com>