**Comprehensive Development of Restful API**

Semester - VIII

## AN INTERNSHIP REPORT

***Submitted by***

# Thakkar Parth Riteshbhai

## 200170107016

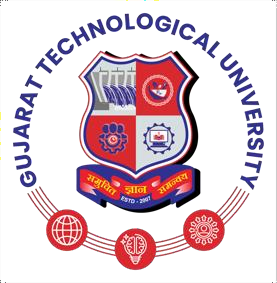
***In partial fulfilment for the award of the degree of***

# BACHELOR OF ENGINEERING

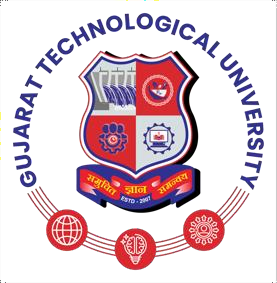
***in***

## Computer Engineering

**VISHWAKARMA GOVERNMENT ENGINEERING COLLEGE CHANDKHEDA**



# Gujarat Technological University, Ahmedabad April, 2024

# Vishwakarma Government Engineering College Chandkheda, Ahmedabad, 382424

CERTIFICATE

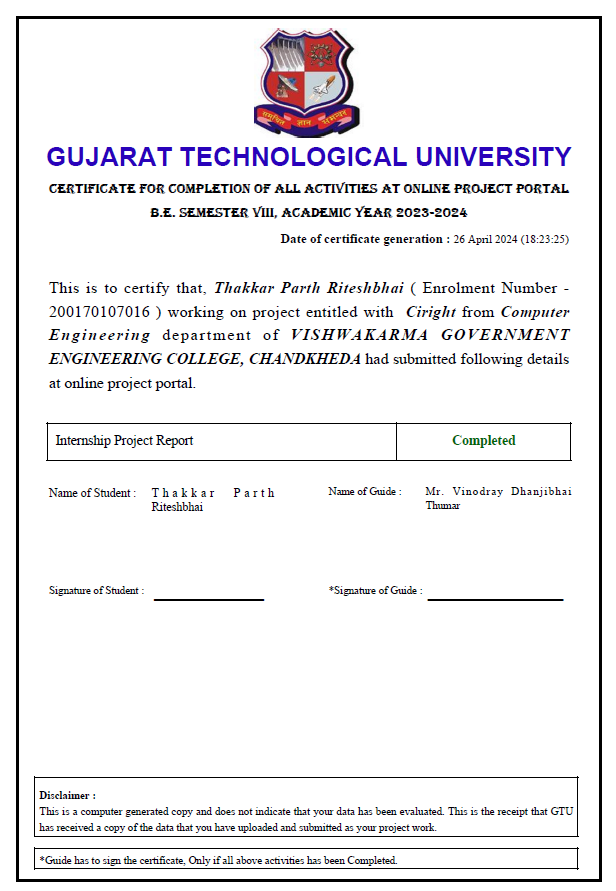
This is to certify that the Internship report submitted along with the entitled **Comprehensive Development of Restful API** has been carried out by **Thakkar Parth Riteshbhai (200170107016)** under my guidance in partial fulfilment of the degree of BE in Computer Engineering (8th Semester) of Gujarat Technological University, Ahmedabad during the academic year 2023-24.

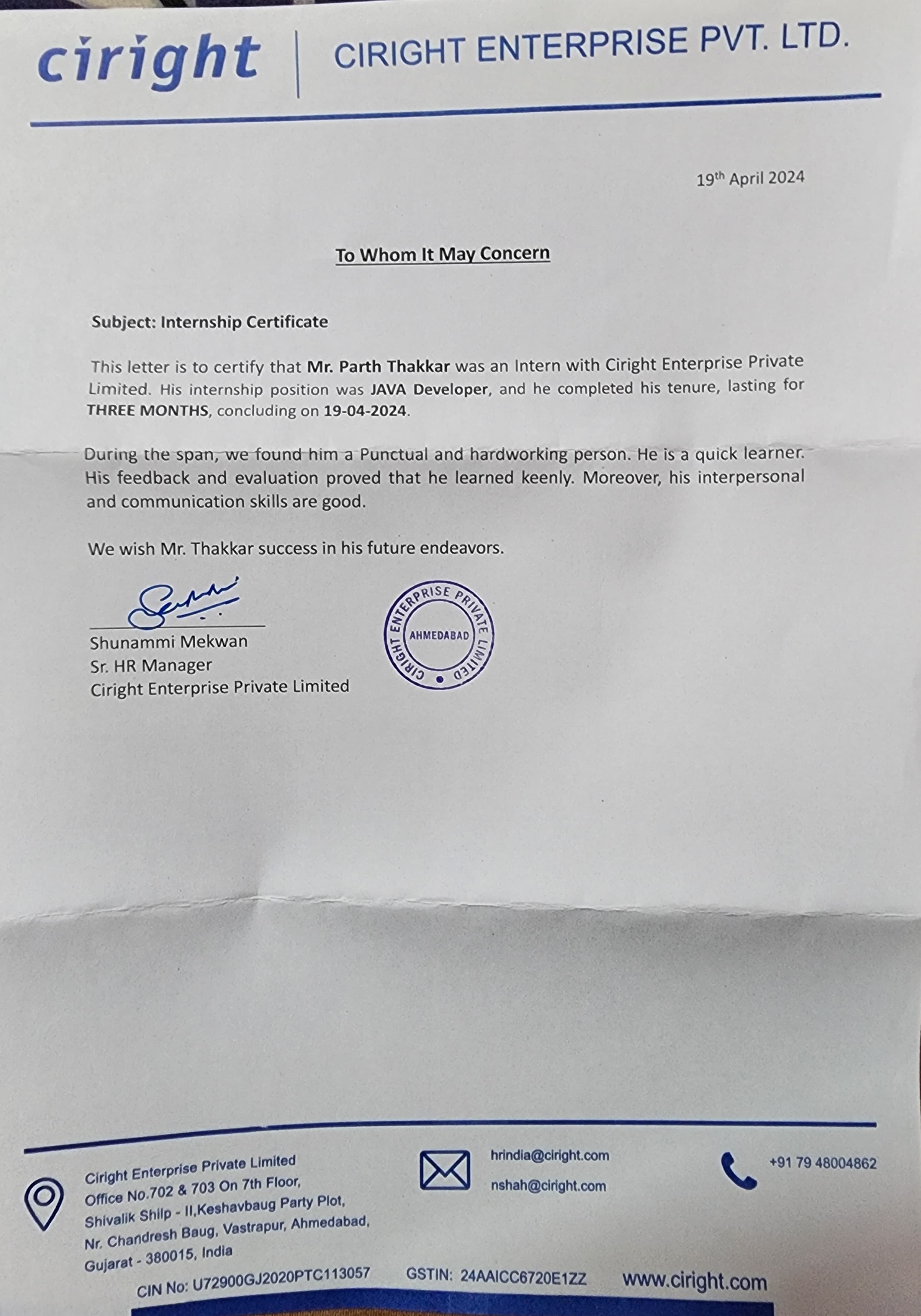
**Internal Guide Head of Department**

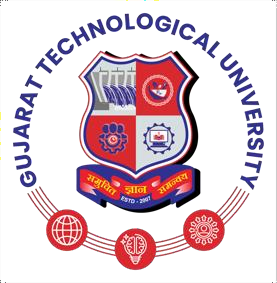
Prof.Vinod D Thumar Prof. Kajal S. Patel

Assistant Professor Associate Professor

VGEC Chandkheda VGEC Chandkheda





DECLARATION

I hereby declare that the Internship report submitted along with the Internship entitled **Comprehensive Development Of Restful API** submitted in partial fulfilment for the degree of Bachelor of Engineering in Computer Engineering to Gujarat Technological University, Ahmedabad, is a bonafide record of original Internship work carried out by me at **Ciright Enterprise Private Limited** under the supervision of **Mrs.** **Shunammi Mekwan** and that no part of this report has been directly copied from any student’s reports or taken from any other source, without providing due reference.

Name of the Student Sign of Student

Thakkar Parth Riteshbhai

# ACKNOWLEDGEMENT

We would like to extend heartiest thanks to Mr. Ankit Mekwan (Tech Lead at Ciright) for supporting us during the internship period. He guided us all the time and motivated us within his busy schedule.

We heartily thank Prof. Kajal S. Patel (Head, Computer Engineering Department) for providing us the opportunity and giving us exposure to pursue our final semester internship at “Ciright Enterprise Private Limited” and gain practical working experience from the industry.

Our sincere thanks to Prof. Vinod D Thumar (Computer Engineering Department) for the unconditional and overwhelming support during the entire session of study and development, also for guiding us throughout the internship period. All the faculties from our department provided us a favourable environment and necessary guidance, without them we would not have achieved our goal. They have always been available for us despite their busy schedule and were always a great source of inspiration for us.

A blend of gratitude, pleasure and great satisfaction is what we feel to convey our indebtedness to all those who have directly and indirectly contributed to the successful completion of the project.

Thanking you,

**Thakkar Parth**

# ABSTRACT

In the dynamic landscape of modern web development, the construction of robust Application Programming Interfaces (APIs) stands as a cornerstone for facilitating seamless communication between diverse software systems. This project delves into the realm of API development, leveraging the powerful combination of Spring MVC and Hibernate frameworks to create efficient and scalable RESTful APIs.

Within the scope of this internship at Ciright Enterprise Private Limited, I have undertaken the task of crafting a suite of APIs to fulfill various HTTP methods, encompassing GET, POST, DELETE, PUT, and validation functionalities. These APIs serve as the bridge between the front-end applications and the underlying data storage, facilitating the retrieval, creation, modification, and deletion of resources in a structured and controlled mann

# List of Figures

[Figure 1.1: Company Logo](#_bookmark19) 1

[Figure 2.1: Gantt Chart](#_bookmark21) 9

[Figure 4.1: Use Case Diagram](#_bookmark41) 15

[Figure 4.2: Activity Diagram](#_bookmark43) 16

[Figure 4.3: DFD Level 0 Diagram](#_bookmark46) 17

[Figure 4.4: Structure Design](#_bookmark48) 17

[Figure 4.5: Flowchart](#_bookmark50) 18

[Figure 5.1: GET Request Method](#_bookmark52) 19

[Figure 5.2: POST Request Method](#_bookmark54) 20

[Figure 5.3: DELETE Request Method](#_bookmark66) 21

[Figure 5.4: PUT Request Method](#_bookmark67) 22

[Figure 5.5: Validate Order Number](#_bookmark68) 23

# List of Tables

[Table 6.1: GET Request Method](#_bookmark86) 26

[Table 6.2: POST Request Method](#_bookmark87) 27

[Table 6.3: DELETE Request Method](#_bookmark88) 28

[Table 6.4: PUT Request Method](#_bookmark89) 29

[Table 6.5: Validate Method](#_bookmark90) 30

# Table of Contents

[ACKNOWLEDGEMENT I](#_bookmark0)

[ABSTRACT II](#_bookmark1)

[LIST OF FIGURES III](#_bookmark2)

[LIST OF TABLES IV](#_bookmark3)

[TABLE OF CONTENTS](#_bookmark5) V

1. [**OVERVIEW OF COMPANY 1**](#_bookmark6)
   1. [COMPANY PROFILE 1](#_bookmark7)
   2. [COMPANY PRODUCTS 1](#_bookmark8)
   3. [COMPANY MISSION AND VISION](#_bookmark9) 2
2. [**INTRODUCTION TO INTERNSHIP AND INTERNSHIP**](#_bookmark12)[**MANAGEMENT 3**](#_bookmark12)
   1. [INTERNSHIP SUMMARY 3](#_bookmark13)
   2. [PURPOSE 3](#_bookmark14)
   3. [OBJECTIVE 4](#_bookmark15)
   4. [TECHNOLOGY AND LITERATURE REVIEW](#_bookmark16) 5
   5. [INTERNSHIP PLANNING](#_bookmark17) 7
      1. [INTERNSHIP DEVELOPMENT APPROACH AND](#_bookmark18) [JUSTIFICATION.](#_bookmark18) 7

2.5.2 ROLES AND RESPONSIBILITIES 8

2.6 [INTERNSHIP SCHEDULING (GANTT CHART)](#_bookmark20) 9

1. [**SYSTEM ANALYSIS**](#_bookmark22) **10**
   1. [STUDY OF CURRENT SYSTEM](#_bookmark23) 10
   2. [PROBLEM AND WEAKNESSES OF CURRENT SYSTEM](#_bookmark24) 11
   3. [REQUIREMENTS OF NEW SYSTEM](#_bookmark25) 11
   4. [SYSTEM FEASIBILITY](#_bookmark26) 12
      1. [TECHNICAL FEASIBILITY:](#_bookmark27) 12
      2. [TIME SCHEDULE FEASIBILITY](#_bookmark28) 12
      3. [OPERATIONAL FEASIBILITY](#_bookmark29) 12
      4. [DOES THE SYSTEM CONTRIBUTE TO THE OVERALL OBJECTIVES OF](#_bookmark31) [THE ORGANIZATION?](#_bookmark31) 12
      5. CAN THE SYSTEM BE IMPLEMENTED USING THE CURRENT TECHNOLOGY AND WITHIN THE GIVEN COST AND SCHEDULE CONSTRAINTS? 13
      6. [CAN THE SYSTEM BE INTEGRATED WITH OTHER SYSTEMS WHICH](#_bookmark33) [ARE ALREADY IN PLACE?](#_bookmark33) 13
   5. [FEATURES OF PROPOSED SYSTEM](#_bookmark34) 13
   6. [LIST OF MAIN COMPONENTS](#_bookmark35) 13
   7. [SELECTION OF HARDWARE / SOFTWARE](#_bookmark36) 13
2. [**SYSTEM DESIGN**](#_bookmark37) **14**
   1. [SYSTEM DESIGN AND METHODOLOGY](#_bookmark38) 14
   2. [ARCHITECTURAL DIAGRAMS](#_bookmark39) 15
      1. [USE CASE DIAGRAM](#_bookmark40) 15
      2. [ACTIVITY DIAGRAM](#_bookmark42) 16
      3. [DFD LEVEL 0 DIAGRAM (CONTEXT DIAGRAM)](#_bookmark45) 17
      4. [STRUCTURE DESIGN](#_bookmark47) 17
      5. [FLOWCHART](#_bookmark49) 18
3. [**IMPLEMENTATION**](#_bookmark63) **19**
   1. [IMPLEMENTATION PLATFORM](#_bookmark64) 19
   2. [MODULE SPECIFICATIONS](#_bookmark65) 19
4. [**TESTING**](#_bookmark83) **24**
   1. [TESTING STRATEGIES](#_bookmark84) 24
   2. [TESTCASES](#_bookmark85) 26
5. [**CONCLUSION AND DISCUSSION**](#_bookmark94) **31**

8.1 OVERALL ANALYSIS OF INTERNSHIP VIABILITIES 31

8.2 DATE OF CONTINUOUS EVALUATION 32

8.3 PROBLEM ENCOUNTERED AND POSSIBLE SOLUTIONS 32

8.4 SUMMARY OF INTERNSHIP WORK 33

[**REFERENCES**](#_bookmark95) **34**

**APPENDIX 35**

# CHAPTER 1. OVERVIEW OF COMPANY

## COMPANY PROFILE

## Figure 1.1: Company Logo

## Ciright Enterprise is a USA based company with a branch in Ahmedabad that focuses on how to make a world more digitally equipped. It is a product based company that makes product for human free zone and betterment of the technology. It makes the changes in the world of science by making the products that are consumed by human for them to function properly with well being Ciright Enterprise Pvt Ltd. team of experienced professionals is dedicated to delivering high-quality services that meet the unique needs of each client.From web design and development to living applications models that brings the digital world live, the company offers a wide range of solutions that are designed to drive results.In addition to its core services, Ciright Enterprise also offers custom analytics and reporting solutions that help businesses to make informed decisions and stay ahead of the competition. With a focus on collaboration, transparency, and innovation, the company is committed to delivering exceptional results that exceed clients' expectations. Overall, Ciright Enterprise is a trusted partner for businesses looking to succeed in today's digital and technology landscape. With its expertise, experience, and innovative approach, the company is well-positioned to help businesses of all sizes to achieve their application development goals.

## COMPANY PRODUCTS

Ciright Enterprise Pvt Ltd. provides a range of products and services related to

webdesign and the application development. It moreover makes the products

that can make the world more finite for a better use of the technology. They

offer customizable website templates for various industries and management

services to increase brand visibility and engagement. They make the company

products that are able to enhance the data of the services and create a verse of

technology to perform better for them.

It's main focus is to provide a product with they make on their own for the

clients to buy to retain their helpful service and makes the human work better.

It makes the products for entertainment, data recovery, IT services, security

purpose to achieve, Fintech, Edtech, spatial computing and better meta

versing of digital world.

## COMPANY MISSION AND VISION

Mission: To connect the physical world to the virtual world through AR VR and mixed realities, for the future digital world to grow more better. It also focuses on the connecting the IOT technologies from the beer tape to make them the real world. Ciright core is the game changer for every other in the living verse meeting the digital enterprise and the engine.

Vision: The main focus of the Ciright Enterprise is to trailblazer in spatial computing, seamlessly integrating virtual, augmented and mixed and mixed realities to create immersive and interactive digital experiences. Our expertise extends to metaverse technology, where we empower business to thrive evolving digital landscape.

## 

# CHAPTER 2. INTRODUCTION TO INTERNSHIP AND INTERNSHIP MANAGEMENT

## INTERNSHIP SUMMARY

During my three-month internship, I embarked on a journey of learning and growth, honing my skills in various tools, technologies, and project management practices essential for modern software development.

Before diving into the live project, I decided to shore up my basics, taking advice from experienced Java developers. I started from scratch, getting comfortable with Java's basic concepts. Then, I moved on to learning about client-server communication, servlets, annotations, and the like.

Once I felt confident with those, I tackled one of the main tasks for my internship: learning Spring MVC. I spent time understanding how Controllers and Models work within the MVC architecture. It was all about getting to grips with how they functioned and how they interacted.

With that knowledge under my belt, I put theory into practice. Using Spring MVC, I set out to develop APIs for different HTTP request methods, making sure to follow best practices and design principles. After implementation, I thoroughly tested the APIs to ensure they worked as expected and performed well.

This journey, from learning the basics to building and testing APIs, was a gradual but rewarding process that showcased my growth and learning throughout the internship.

## PURPOSE

The primary objective of my internship was to acquire practical experience and deepen my understanding of software development, particularly in the realm of web application development using Java technologies. With a foundational background in Java programming, I sought to expand my skill set and knowledge base by delving into real-world projects under the guidance of experienced professionals.

The purpose of my internship journey was threefold:

1. **Knowledge Enhancement**: I aimed to strengthen my foundational knowledge of Java

programming and broaden my understanding of related technologies such as Spring MVC and Hibernate. By immersing myself in the details of these frameworks, I aimed to strengthen my proficiency in developing robust and scalable web applications.

1. **Hands-on Experience**: Armed with theoretical knowledge, my goal was to translate concepts into practical applications by actively participating in the development lifecycle of real-world projects. Through hands-on experience, I aimed to sharpen my problem-solving skills, gain exposure to industry best practices, and cultivate a pragmatic approach to software development.
2. **Professional Growth**: Beyond technical skills, I aspired to cultivate essential soft skills such as teamwork, communication, and adaptability. By collaborating with seasoned professionals and working within a dynamic team environment, I aimed to refine my interpersonal skills and adapt to the demands of professional software development.

## OBJECTIVE

During my internship, I set out to achieve the following objectives:

1. **Deepen Understanding of Java Technologies**: My primary objective was to deepen my understanding of Java programming language and its associated frameworks, particularly Spring MVC and Hibernate. By immersing myself in practical projects and guided learning sessions, I aimed to enhance my proficiency in Java development and gain hands-on experience with industry-standard tools and methodologies.
2. **Improving Logic on API development**: By creating different API as a part of pratice, I sought to master the detailed understanding of the logic and relation between different components in the making of API such as controller, model, POJO file, etc. Through focused study and practical application, my objective was to gain a comprehensive understanding of the concepts like Annotations and Request Mapping.
3. **Develop Real-World Application Skills**: With a keen focus on practical learning, I aimed to develop real-world application development skills by actively participating in the development lifecycle of projects. My objective was to apply theoretical knowledge to solve practical challenges, design and implement scalable solutions, and adhere to

industry best practices in software engineering.

1. **Enhance Collaboration and Communication Skills**: Recognizing the importance of teamwork and effective communication in professional settings, I set out to enhance my collaboration skills during the internship. My objective was to actively engage with team members, seek and provide constructive feedback, and contribute positively to team dynamics, encouraging a conducive environment for collective growth and success.
2. **Gain Exposure to Industry Practices**: I wanted to learn more about how things work in the industry. By working in the internship, I hoped to understand what's happening now and what's new in software development. My goal was to keep up with what's happening in the field and always be ready to learn new things.

By aligning my efforts with these objectives, I aimed to maximize my learning experience, contribute meaningfully to project outcomes, and lay a solid foundation for my future career aspirations in software engineering.

## TECHNOLOGY AND LITERATURE REVIEW

## 2.4.1 Technology:

Throughout the course of my internship, I had the opportunity to work witha diverse range of technologies, enabling me to gain valuable hands-on experience and enhance my skill set. The key technologies utilized in my internship project included:

* + 1. **Java:**

As the primary programming language, Java served as the foundation for my development efforts. I leveraged Java's robust features and libraries to implement various functionalities within the project.

### 

* + 1. **Spring MVC:**

A key component of the project, Spring MVC facilitated the development of web applications by providing a comprehensive framework for building scalable and maintainable solutions. I utilized Spring MVC to implement controllers, handle HTTP requests, and manage application flow.

* + 1. **Hibernate**:

Hibernate played a pivotal role in simplifying data access and persistence within the project. By providing an object-relational mapping (ORM) framework, Hibernate abstracted away the complexities of database interactions, allowing for seamless integration with Java objects.

### RESTful API Development:

### As the focus of the project revolved around building RESTful APIs, I gained proficiency in designing and implementing APIs that adhere to REST principles. This involved utilizing HTTP methods (GET, POST, PUT, DELETE), defining resource endpoints, and ensuring proper request and response handling.

* + 1. **Database Management Systems (DBMS):**

I worked with MySQL as the underlying database management system for storing and retrieving application data. This involved schema design, SQL query optimization, and data manipulation tasks.

* + 1. **Version Control System:**

Collaboration and version management were facilitated through Git. I leveraged Git for tracking changes, managing code repositories, and coordinating with team members.

* + 1. **API Testing Tool:**

I utilized the "Talend API Tester" as a versatile tool for testing and validating the functionality of RESTful APIs developed during the internship. This intuitive extension provided a user-friendly interface for crafting HTTP requests, inspecting responses, and troubleshooting API endpoints. Its robust features and seamless integration with Chrome browser proved instrumental in ensuring the reliability and effectiveness of the developed APIs.

**2.4.2 Literature Review:**

Development of API is one of the major task in maintaining the flow of the live project named Ciright. A thorough understanding of Spring MVC and Database

Concepts is essential. Whereas the basic concepts of Java act as foundation. CRUD operations are the crucial ones in the API development. Thus, building a Web Spring MVC project as the initial step and implementing various concepts such as CRUD, multiple controller, usage of various Annotations, etc along with Database created from scratch played a very important role.

## INTERNSHIP PLANNING

* + 1. **Internship Development Approach and Justification:**

Throughout the duration of my internship, I adopted a structured and iterative approach to development, leveraging a combination of self-directed learning, practical application, and collaborative problem-solving. The key components of my development approach included:

* + 1. **Learning and Familiarization:**

I began by immersing myself in foundational learning, taking guidance from senior Java developers and self-paced online resources. This phase focused on gaining familiarity with core Java concepts, client-server communication, servlets, and other relevant technologies essential for web application development.

* + 1. **Focused Study of Spring MVC and Hibernate:**

Recognizing the significance of Spring MVC and Hibernate in the context of my internship project, I dedicated considerable time to studying these frameworks in-depth. This involved comprehensive reading of documentation, tutorials, and practical examples to grasp fundamental concepts such as inversion of control, dependency injection, object-relational mapping, and MVC architecture.

* + 1. **Hands-on Practice**:

Building upon theoretical understanding, I transitioned towards practical application by implementing CRUD operations and RESTful APIs using Spring MVC and Hibernate. This hands-on practice allowed me to reinforce concepts learned and gain proficiency in applying frameworks to real-world development scenarios.

* + 1. **Iterative Development and Testing**:

I adopted an iterative approach to development, breaking down tasks into manageable chunks and prioritizing deliverables based on project requirements. Continuous testing and validation of implemented functionalities ensured early detection of bugs and adherence to specifications, facilitating timely iterations and refinements.

* + 1. **Collaborative Problem-solving**:

Throughout the development process, I actively sought guidance from mentors, collaborated with team members, and participated in discussions to address challenges and explore solutions collaboratively. This collaborative problem-solving approach fostered knowledge sharing, facilitated peer learning, and enriched the overall internship experience.

* + 1. **Adaptation and Continuous Improvement**:

As the project evolved and requirements evolved, I remained flexible and adaptive, embracing change and adjusting my development approach accordingly. Regular retrospectives and self-reflection enabled me to identify areas for improvement and implement corrective measures, fostering a culture of continuous learning and growth.

## Roles and Responsibilities:

## As it was my learning phase, almost all the things related to API was my responsibility. Such as understanding why a particular API is being asked, then logic to build that API. Once it is created, testing it and assigning a unique number to it was also my task.

## INTERNSHIP SCHEDULING (Gantt Chart)

## The figure 2.1 below shows the gantt chart of my overall internship. The duration was 13 weeks, starting from 19th January to 19th April, involving two main subsection: Training and Working in live project. The Arrows between task shows their inter-dependency.



Figure 2.1: Gantt Chart

# CHAPTER 3. SYSTEM ANALYSIS

## STUDY OF CURRENT SYSTEM

## Analysis of Web Application Development Landscape: I researched and analyzed the prevailing trends, methodologies, and frameworks used in web application development. This involved studying industry reports, articles, and case studies to identify emerging technologies, adoption trends, and key challenges faced by developers.

* **Evaluation of RESTful API Development Practices:** As RESTful APIs formed the core focus of my internship project, I examined current practices and standards for designing, implementing, and testing APIs. This included reviewing API documentation, studying API design principles, and analyzing real-world examples to identify common patterns and best practices.
* **Assessment of Spring MVC and Hibernate Frameworks:** Given the centrality of Spring MVC and Hibernate in my project, I conducted a detailed evaluation of these frameworks. This involved studying official documentation, tutorials, and community forums to understand their features, capabilities, and recommended usage patterns. Additionally, I explored case studies and success stories to gain insights into real-world implementations and performance considerations.
* **Identification of Potential Areas for Enhancement:** Based on my analysis of the current system and industry practices, I identified potential areas for enhancement and optimization. This included pinpointing opportunities to leverage advanced features or extensions of Spring MVC and Hibernate, streamline development workflows, and improve the overall efficiency and scalability of the system.
* **Documentation and Reporting:** Throughout the study, I maintained detailed documentation of my findings, including summaries of key insights, references to relevant resources, and actionable recommendations for future development efforts. This documentation served as a valuable reference point for informing decision-making and guiding the subsequent phases of the internship project.

## PROBLEM AND WEAKNESSES OF CURRENT SYSTEM

* **Limited Support for Versioning and Compatibility:** The existing APIs do not adequately support versioning and backward compatibility, making it challenging to introduce changes or enhancements without disrupting existing consumers. This lack of versioning mechanisms results in dependencies on specific API versions and impedes the adoption of new features or improvements.
* **Insufficient Error Handling and Response Validation:** The current system lacks robust error handling mechanisms and response validation techniques, resulting in inadequate feedback to clients and potential security vulnerabilities. Improper error handling can lead to confusion, data integrity issues, and exposure to malicious attacks, undermining the reliability and security of the APIs.

## REQUIREMENTS OF NEW SYSTEM

* **Standardized API Design:** The new system should adhere to standardized API design principles and conventions, promoting consistency, clarity, and ease of use across all endpoints. Well-defined resource naming conventions, consistent error handling mechanisms, and clear documentation should be established to facilitate developer understanding and adoption.
* **Versioning and Compatibility Support:** The new system should incorporate robust versioning mechanisms to support backward compatibility and seamless evolution of APIs over time. Versioning strategies such as URI versioning, media type versioning, or header-based versioning should be implemented to allow for independent evolution of API versions and smooth migration paths for consumers.
* **Enhanced Error Handling and Validation:** The new system should implement comprehensive error handling mechanisms and response validation techniques to ensure reliable and secure API interactions. Proper error codes, informative error messages, and consistent error formats should be provided to facilitate effective troubleshooting and debugging by consumers.
* **Comprehensive Testing and Documentation:** The new system should undergo comprehensive testing to validate its functionality, reliability, and security. Unit tests,

integration tests, and end-to-end tests should be conducted to verify the correctness of API behaviors and interactions. Additionally, thorough documentation should be provided to guide developers in utilizing the APIs effectively and troubleshooting common issues.

## SYSTEM FEASIBILITY

A feasibility study is an evaluation of a design to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation. In other words, a feasibility study is an evaluation or analysis of the potential impact of a project. As far as this project is concerned here the possibilities and the viability are concerning itself with the techniques, cost analysis, economic analysis, etc. used in these projects which are as follows:

## Technical Feasibility

What we have planned to implement is technically feasible. Do we have sufficient knowledge or technology to make it a reality? And the answer is easy because all I have to perform in IDE and any other required software is available in marketplace of that IDE. We have tried to maintain the coordination between developing logic, creating and testing. Now we can conclude that the system is technically feasible.

## Time Schedule Feasibility

In the matter of time, the entire process of creating API is time efficient. Most of the time it has to go through 4 steps that is, the 4 things that collectively can be called as API. The core component is Controller then an interface and its implementation file and lastly Model file to store fetched data. So, summing up, it depends on the complexity and expectations of an API and experience of a developer.

## Operational Feasibility

How the project will work and who will use it, all such concerns arise in this phase. We have to study the existing system’s problem and whether it is worth solving or not.

## Does the system contribute to the overall objectives of the organization?

Themain task of the API is to do whatever the client is commanding to do whether it would be of GET, POST or PUT request. So, yes it is one of the core step in fulfilling the organization’s objective.

## Can the system be implemented using the current technology and within the given cost and schedule constraints?

Absolutely, infact the organization is creating APIs since very long time with no additional cost using the same technology they started with that too very efficiently.

## 

## Can the system be integrated with other systems which are already in place?

Yes, as I mentioned earlier that there no changes in the process or technology of creating API so, there will be no compatibility issue neither there will be any conflict.

## FEATURES OF PROPOSED SYSTEM

## Standardized API Design

## Versioning and Compatibility Support

## Enhanced Error Handling and Validation

## Optimized Performance and Scalability

## Simplified Data Persistence and Mapping

## Comprehensive Testing and Documentation

## LIST OF MAIN COMPONENTS

* Controller
* Model
* Service File (Interface)
* ServiceImpl (Implementation)
* POJO File of DataTable

## SELECTION OF HARDWARE / SOFTWARE

## Eclipse IDE

## Maven

## Apache Tomcat

## Talend API tester

## Git

**CHAPTER 4. SYSTEM DESIGN**

## SYSTEM DESIGN AND METHODOLOGY

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

### Architectural Design

The architectural design of a system emphasizes the design of the system's architecture that describes the structure, behavior and more views of that system and analysis.

### Logical Design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included.

### Physical Design

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed. In physical design, the following requirements about the system are decided.

* Input requirement
* Output requirements
* Storage requirements
* Processing requirements

## ARCHITECTURAL DIAGRAM:

## Use case Diagram:

## Actors for support Platform

* + Admin
  + User

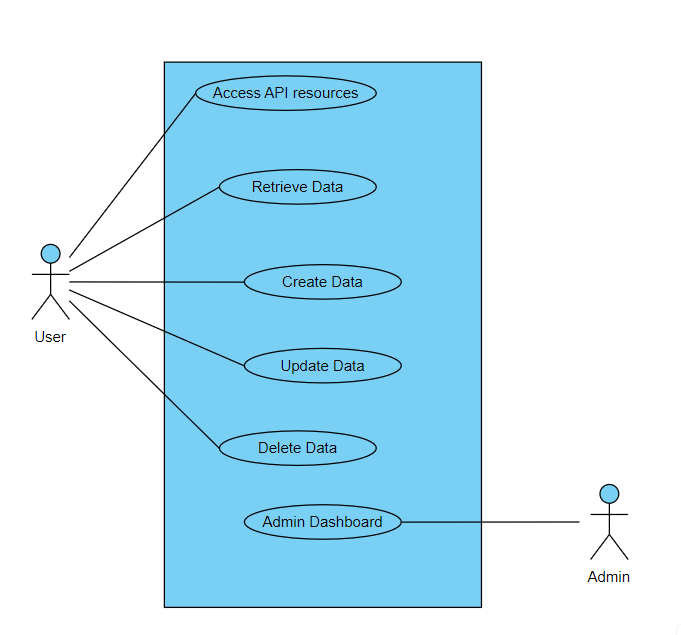
 The use case diagram (figure 4.1) illustrates the interactions between actors and the system, showcasing the functionalities supported by the developed RESTful APIs with Spring MVC and Hibernate. Actors include Users who interact with the APIs to perform actions such as accessing resources, retrieving, creating, updating, and deleting data. Optional actors, like Admins, may have additional privileges or access levels. The diagram provides a visual representation of the system's capabilities and user interactions.

Figure 4.1: Use Case Diagram

## Activity Diagram:

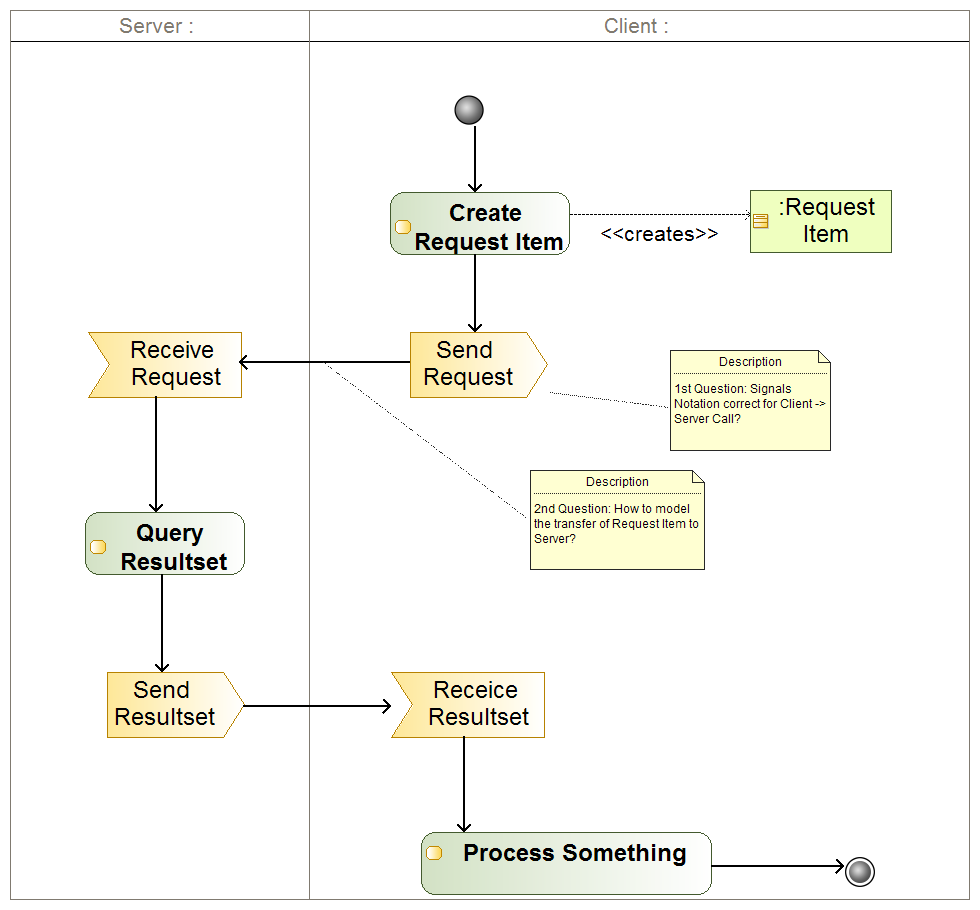
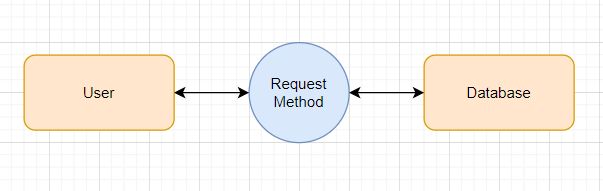
The activity diagram (figure4.2) demonstrates the sequence of actions involved in processing client requests through the RESTful APIs developed using Spring MVC and Hibernate. It outlines the workflow from the initiation of a request to its completion, highlighting key steps such as request parsing, data retrieval or manipulation, database operations, and response generation.

Figure 4.2: Activity Diagram

## DFD Diagram:

The Data Flow Diagram level-0 (DFD level-0) illustrates the flow of data within the system, focusing on how information moves between various components and processes. In the context of developing RESTful APIs with Spring MVC and Hibernate, the DFD can depict the flow of data from client requests through the API endpoints to the backend database and back to the client as responses.



## Figure 4.3: DFD Level-0 Diagram

## Structure Design:

## The figure 4.4 shows the general structure of an API which includes some core components such as Controller, Interface and Model.

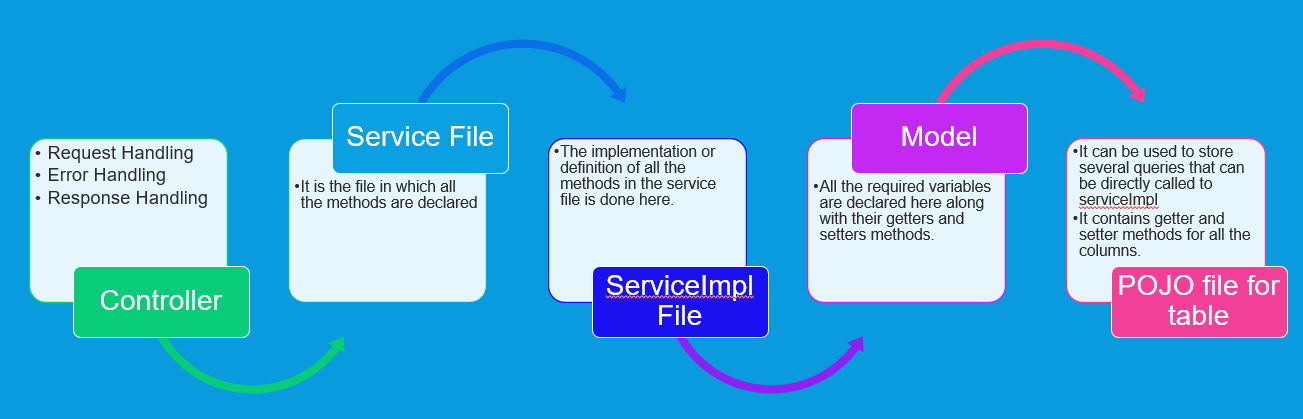


Figure 4.4: Structure Design

## Flowchart:

## The below figure 4.5 is the flowchart of how an API works mentioning some checkpoints such as HTTP Method and Authentication.

Figure 4.5: Flowchart

# CHAPTER 5. IMPLEMENTATION

## IMPLEMENTATION PLATFORM

My work on creating API is completely implemented on Eclipse IDE.

## MODULE SPECIFICATIONS

## GET Request Method:

## The figure 5.1 shows how GET Request Method is implemented. It can be seen that upon correct execution it fetches and displays requested data.

Figure 5.1: GET Request Method

## POST Request Method:

## The implementation of POST Request Method is shown in the figure 5.2, where passing correct data fields, desired data can be added to database.

Figure 5.2: POST Request Method

### DELETE Request Method:

### The implementation of DELETE Request Method is shown in figure 5.3, in which, passing correct ID in URL will delete that data from database.

### 

Figure 5.3: DELETE Request Method

### PUT Request Method:

### The implementation of PUT Request Method is shown in figure 5.4, which can be used to only update the existing data in database by passing respective ID of that data in URL.

Figure 5.4: PUT Request Method

### Validate Order Number:

### The implementation of Validation method is shown in figure 5.5, which will validate that provided order number is unique or not.

Figure 5.5: Validate Order Number

# CHAPTER 6. TESTING

## TESTING STRATEGIES

There are different Models of testing. On the basis of testing methods there are two types of testing:

* + - Black-box testing.
    - White-box testing

Black-box tests are used to demonstrate that software functions are operational, that input is properly accepted and output is correctly produced, and that integrity of external information is maintained.

White-box tests are used to examine the procedural details. It checks the logical paths by test case. It can also check the conditions, loops used in the software coding. It checks that loops are working correctly on defined boundary value.

1. **White-Box Testing:**
   * White-box testing sometimes called glass-box testing, is a test case design method that users the control structure of the procedural design to drive the test case.
   * Always we are thinking that there is no necessary to execute or checks the loops and conditions. And so large number of errors is uncovered. With using white-box testing methods, we have checked that, all independent paths within a function have been executed at least once. All loops working correctly at their boundary values and within their specified conditions.
   * In our coding we test that all the loops work truly in each module. The one technique of white- box testing is basis path testing. It contains two parts; one is flow graph notation and the second is cyclometer complexity. In flow graph notation we are checking logical control of flow. By using cyclometer complexity, we find complexity of our project structure

### 2.) Block-box Testing:

* + Black-box testing focuses on the functional requirements of the software. That is black- box testing enables the software engineer to drive sets of input conditions that will fully exercise all functional Requirements for the program. Black-box testing is not an alternative to white-box testing techniques. Rather, it is a complementary approach that is likely to uncover a different class of errors than white-box methods.
  + We use in our coding to find errors in the following categories:
  + Incorrect or missing functions
  + Interface errors
  + Errors in database
  + Performance errors
  + Initialization and termination errors.
  + Unlike white-box testing, which is performed earlier in the testing process, black-box testing tends to be applied during later stages of testing. Because black-box testing purposely disregards control structure, attention is focused on the information domain. By applying black-box techniques, we derive a set of test cases that satisfy following criteria. Test cases that reduce, by a count that is greater than one, the number of additional test cases must be designed to achieve reasonable testing. Test cases that tell us something about the present or absence of classes of errors, rather than an error associated only with the specific test at hand.

## TEST CASES

## Below table 6.1 are the test cases for GET Request Method to check its implementation in various conditions such as correct parameter in URL and JSON format.

Table 6.1: GET Request Method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test case description** | **Input data** | **Expected Result** | **Actual Result** | **Pass/**  **Fai l** |
| TS\_1 | For GET method, the parameters are passed in the URL itself. Do this and press the send button. | Parameter in the URL | If required parameters are passed then data is fetched. | As Expected | Pass |
| TS\_2 | Wrong parameters are provided in the URL. | Parameter in the URL | It throws the error of bad request. | As Expected | Pass |
| TS\_3 | No parameters are passed in the URL. | URL without parameter | It throws the error of bad request. | As Expected | Pass |

## Below table 6.2 are the test cases for POST Request Method to check its implementation in various conditions such as missing data fields and correct JSON format.

Table 6.2: POST Request Method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test case description** | **Input data** | **Expected Result** | **Actual Result** | **Pass/**  **Fail** |
| TS\_1 | Data fields to be added in database is passed externally in POST method in JSON format. | All the data that user wants to add including certain mandatory fields. | The passed data would get successfully added in the database. | As Expected | Pass |
| TS\_2 | Data fields should be in correct json format. | All the data that user wants to add including certain mandatory fields with incorrect json format. | An error will be thrown of internal server error. | As Expected | Pass |
| TS\_3 | All the mandatory data fields should be passed with correct value. | Some mandatory fields were not passed or were incorrect. | An error will be thrown of Bad Request. | As Expected | Pass |

## Below table 6.3 are the test cases for DELETE Request Method to check its implementation in various conditions such as correct parameter in URL.

Table 6.3: DELETE Request Method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test case description** | **Input data** | **Expected Result** | **Actual Result** | **Pass/ Fail** |
| TS\_1 | Id of the data that user wants to delete should be passed in the URL | Respective Id in URL | Data will be deleted from database. | As expected | Pass |
| TS\_2 | Id of the data that user wants to delete should be passed in the URL | No Id was provided. | It throws the error of bad request. | As expected | Pass |

## Below table 6.4 are the test cases for PUT Request Method to check its implementation in various conditions such as correct parameter in URL and JSON format.

Table 6.4: PUT Request Method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test case description** | **Input data** | **Expected Result** | **Actual Result** | **Pass**  **/Fail** |
| TS\_1 | Id of the data that user wants to update should be passed in the URL | Respective Id in URL | Data will get updated in database. | As expected | Pass |
| TS\_2 | Id of the data that user wants to update should be passed in the URL | No Id was provided. | It throws the error of bad request. | As expected | Pass |
| TS\_3 | Data fields to be updated in database is passed externally in POST method in JSON format. | All the data that user wants to update including certain mandatory fields. | The data will get successfully updated in the database. | As expected | Pass |
| TS\_4 | Data fields should be in correct json format. | All data that needs to update including mandatory fields with incorrect json format. | An error will be thrown of internal server error. | As Expected | Pass |
| TS\_5 | All the mandatory data fields should be passed with correct value. | Some mandatory fields were not passed or were incorrect. | An error will be thrown of Bad Request. | As Expected | Pass |

## Below table 6.5 are the test cases for Validate Method to check its implementation in various conditions such as Unique order number or missing data fields.

Table 6.5: Validate Method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test case description** | **Input data** | **Expected Result** | **Actual Result** | **Pass/**  **Fai l** |
| TS\_1 | Order no. to be validated is passed along with some mandatory data fileds. | Unique order number with other data fields. | Order no. will get validate and will be assign to the Id passed along with it. | As expected | Pass |
| TS\_2 | Order no. to be validated is passed along with some mandatory data fileds. | Already used order number with other data fields. | Order number already exist will be prompted in the result section. | As expected | Pass |
| TS\_3 | Data fields should be in correct json format. | All the data fields with incorrect json format | An error will be thrown of internal server error. | As expected | Pass |

# CHAPTER 7. CONCLUSION AND DISCUSSION

# 7.1 OVERALL ANALYSIS OF INTERNSHIP VIABILITIES

Reflecting on my internship experience, I have gained invaluable insights into the realm of software development and the intricacies of building RESTful APIs using Spring MVC and Hibernate. The internship provided me with a platform to apply theoretical knowledge gained from academic studies to real-world projects, fostering holistic growth and skill development.

* **Knowledge and Skills Acquired:**
* Expanded Understanding of Java: Through hands-on experience and mentorship, I deepened my understanding of Java programming language concepts and best practices.
* Proficiency in Spring MVC: The internship enabled me to master Spring MVC framework, empowering me to develop robust and scalable web applications.
* Expertise in Hibernate ORM: I honed my skills in Hibernate ORM, learning to efficiently map Java objects to database tables and streamline data persistence operations.
* API Development Proficiency: I developed proficiency in designing, implementing, and testing RESTful APIs, encompassing various HTTP request methods and data validation techniques.
* **Challenges and Learning Opportunities:**
* Overcoming Technical Hurdles: I encountered challenges in resolving technical issues related to API development, database interactions, and troubleshooting. However, each challenge presented an opportunity for learning and growth.
* Adaptation to Industry Practices: Adapting to industry-standard practices and methodologies, such as Agile development, version control with Git, and collaborative teamwork, was a learning curve that I successfully navigated.
* **Achievements and Contributions:**
* Successful API Implementation: I successfully designed and developed a range of RESTful APIs to cater to diverse client requirements, implementing CRUD operations,

validation mechanisms, and error handling strategies.

* Collaborative Teamwork: I actively contributed to team discussions, participated in code reviews, and collaborated with colleagues to deliver high-quality solutions within project timelines.
* Positive Feedback and Recognition: My contributions were well-received by peers and supervisors, evidenced by positive feedback and recognition for my dedication, problem-solving skills, and commitment to excellence.

**7.2 DATE OF CONTINOUS EVALUATION (****CE-I and CE-II)**

* **CE-I:**

It was conducted on 16th March by my internal guide, Assistant Professor Karan Bhatt. In First evaluation I showed all the work that I did for company till that date i.e., 8 weeks. I shared each and every experience with him about my training and technology specific training.

* **CE-II:**

Second evaluation was conducted on 6th April. In this evaluation, I presented and posted my internal guide about all the work carried out by me for company after first evaluation.

**7.3** **PROBLEM ENCOUNTERED AND POSSIBLE SOLUTIONS**

* **Technical Challenges:**
* Challenge: Difficulty in resolving technical issues related to API development, database integration, or framework configuration.
* Solution: Seek guidance from mentors, colleagues, or online resources to troubleshoot technical issues effectively.
* **Time Management Constraints:**
* Challenge: Balancing internship responsibilities with academic commitments and personal obligations, leading to time management challenges.
* Solution: Prioritize tasks and allocate time effectively to ensure timely completion of project milestones.
* **Communication and Collaboration Issues:**
* Challenge: Difficulty in coordinating with team members, especially in a remote or distributed work environment, leading to communication gaps or misunderstandings.
* Solution: Utilize collaboration tools and platforms (Google Meet) to facilitate real-time communication and collaboration.
* **Learning Curve Challenges:**
* Challenge: Steep learning curve associated with mastering new technologies, frameworks, or development methodologies.
* Solution: Invest time in self-directed learning and exploration to deepen understanding of key concepts and principles.

**7.4**  **SUMMARY OF INTERNSHIP WORK**

My internship journey was marked by significant growth and learning opportunities in the realm of software development. Throughout this period, I successfully designed and tested various RESTful APIs using Spring MVC and Hibernate, honing my Java programming skills and gaining practical experience in web application development.

Active participation in team discussions, code reviews, and collaborative efforts contributed to the timely delivery of high-quality solutions. I received commendation for my dedication and adaptability, nuturing a culture of excellence within the project environment.

Despite facing challenges such as technical hurdles and time management constraints, I embraced proactive problem-solving strategies. Seeking guidance from mentors and leveraging online resources, I navigated steep learning curves associated with mastering new technologies and frameworks.

Looking ahead, I am inspired to continue my journey in software development, equipped with newfound skills, experiences, and a growth mindset. Grateful for the mentorship received and opportunities provided, I am eager to apply these learnings in future endeavours, contributing to the ever-evolving technology landscape.

# REFERENCES

* + - **Basics of Java**: https://www.javatpoint.com/java-tutorial
    - **Spring MVC**: <https://www.javatpoint.com/spring-mvc-tutorial>
    - **MySQL**: <https://www.w3schools.com/MySQL/default.asp>
    - **RESTful API Design Practices**: <https://www.vinaysahni.com/best-practices-for-a-pragmatic-restful-api>
    - **Gantt Chart:** <https://www.onlinegantt.com/#/gantt>

**APPENDIX**

**PLAGIARISM REPORT**

