Batch Completion Report:

Monocept: JAVA

F.Y. 2023

**December 31, 2023**

**Contents**

[1. Swabhav’s Methodology to Create and Nurture Talent 1](#_Toc150342747)

[a. Student's Learning Journey](#_Toc150342748)

[b. Our Core Philosophy](#_Toc150342749)

[c. How We Teach](#_Toc150342750)

[2. Course Overview](#_Toc150342751) 4

[3. Talent Creation Team 11](#_Toc150342752)

[4. Background of Mentees 12](#_Toc150342753)

[5. Continuous Evaluation of Mentees 13](#_Toc150342754)

[a. Behaviour during the session 13](#_Toc150342755)

[b. Assignments 13](#_Toc150342756)

[6. Project Evaluation 15](#_Toc150342757)

[a. Project Details 15](#_Toc150342758)

[1. Basic Banking App (Servlets/JSP) 15](#_Toc150342759)

[2. Advanced Banking App (REST) 15](#_Toc150342760)

[b. Project Assessment 16](#_Toc150342761)

[7. Growth Report of Batch 18](#_Toc150342762)

[a. Continuous Evaluation: 18](#_Toc150342763)

[b. Project Evaluation 1](#_Toc150342764)9

[c. Overall Growth Analysis of the Batch:](#_Toc150342765) 20

[d. Top Performers](#_Toc150342766) 20

[8. Feedbacks and Observations](#_Toc150342767) 21

[a. Mentors Feedback on Overall batch:](#_Toc150342768) 21

[b. Observations by the Mentor:](#_Toc150342769) 21

[c. Recommendation by the Mentor: 21](#_Toc150342770)

[d. Mentees Feedback about the Course:](#_Toc150342771) 21

# Swabhav’s Methodology to Create and Nurture Talent

## Student's Learning Journey

**Step 1: Talent Enrolment** The journey begins when the company decides to enrol its talent at Swabhav TechLabs.

**Step 2: Creating a Safe Learning Environment** Swabhav establishes a safe learning environment by nurturing mentees' growth through:

* Curiosity
* Surprise
* Positive Error Feedback

**Step 3: Leveraging Neurobiological Drivers** Mentees accelerate their learning with the aid of neurobiological drivers, including:

* Spaced Learning
* Storage Retrieval
* Interleaving

**Step 4: Simplifying Concept Clarity and Syntax Learning** The once challenging steps of achieving concept clarity and mastering syntax become as simple as a hop, skip, and jump.

**Step 5: Soaring into Mastery** Mentees' learning takes flight, guided by a modern system incorporating:

* Projects
* Experiments
* Assignments

## Our Core Philosophy

Our core philosophy revolves around the concept of activated knowledge directly influencing real-world outcomes, embodied in the mantra "Men et Manus," a concept akin to MIT's approach.

## How We Teach

Our teaching methodology focuses on:

* Cultivating neural pathways of trust, respect, curiosity, and abundance to foster exceptional talent.
* Emphasizing experiential learning, centred around creating industry-level projects and experiences that empower mentees to excel.

# Course Overview

**Company**: Monocept

**Course Objectives**:

At the end of the course the mentees will be able to:

1. Gain a fundamental understanding of core and advanced Java and implementation of OOP concepts with more focus on good programming practices and design principles.
2. Understanding HTML, CSS, JavaScript for front end development and learning how to create dynamic and responsive user interfaces using Angular
3. Developing back end applications using Spring Boot and implementing RESTful APIs.
4. Making use of JPA/Hibernate to integrate databases and understand overall database design principles and normalization
5. Integrating front end (Angular) and back end (Java) to create a seamless full stack application
6. Implement best security practices for authentication and authorization mechanism
7. Writing unit test cases with JUNIT to debug, identify and fix the issues.
8. Learning coding standards and industry best practices for writing clean, maintainable and scalable code.

**Date of Commencement**: July 03, 2023

**Date of Completion**: December 30,2023

**Total No of Contact Hours (Mentor with direct contact of Mentees)** : 220

**No of Non-Contact Hours (Evaluation, Project review, Code review, Demos, doubt solving etc..)**: 60

**Modules included**: Core Java, OOAD, Design Patterns, RDBMS/JDBC, JSP Servlets, Spring Core, Spring Boot, Javascript, React, MongoDb, Human Engineering

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.No.** | **Module** | **Topics** | **Sub Topics** |
| 1 | Core Java | Overview of training | JRE/ JDK difference |
| Setting class path |
| Introduction to JAVA Programming | JAVA first simple Program |
| Program structure |
| Installations (JDK, Eclipse, Notepad++) | Basic naming conventions |
| Class file and bytecode |
| Features of Java |
| Compiler and Interpreter |
| Variables, Keywords, Datatypes, final keyword | conditional statements- if, if-else, if-else ladder, switch |
| Accepting Input from command line arguments | integrating eclipse and java API source code |
| Scanner class | Scanner class |
| Control Structures | Control Structures |
| Iterative statements | while, do-while, for, enhanced for |
| Debugging, Call Stack | Debugging, Call Stack |
| Typecasting | Typecasting |
| Wrapper Classes | Wrapper Classes |
| Arrays | Types of Arrays - Single Dimensional, Multidimensional |
| I/O Streams | Inputstreams |
| Outputstreams |
| I/O Streams | Class Diagrams |
| Strings | String, StringBuffer, StringBuilder, Mutable and unmeltability |
| OOP Introduction - Classes | Classes - data members and member methods |
| Objects | Objects |
| Objects | Heap and Stack memory |
| Data Encapsulation | Access Specifiers - private, default, public |
| Getters and setters |
| Constructors | Types of constructors - default, parameterized |
| Garbage Collection | This keyword |
| Passing by value and reference | Memory reclaim |
| finalize() |
| Constructor Chaining | Array of objects |
| Object Calisthenics | Rules 1 , 2 and 6 of Object Calisthenics |
| Enum | Enum |
| Method overloading | Method overloading |
| Refactoring | DRY |
| Refactoring | Reduce else blocks |
| Jar creation | Execute jar file in different platforms windows/linux |
| Static keyword | static methods/variables |
| Static keyword | instance vs static methods |
| Reflection | static blocks execution |
| Inheritance | Inheritance basics |
| Inheritance | extends keyword |
| Types of inheritance - single, multilevel, hierarchical, hybrid |
| Inheritance | Object class - equals(), toString() |
| inheritance with constructors | super keyword uses |
| this() and super() difference |
| Static block and constructor calling | Static block and constructor calling |
| Method overriding | Method overriding |
| Inheritance and polymorphism concept implementation | . polymorphism via inheritance |
| abstract classes implementation | polymorphism via abstract classes |
| interfaces and polymorphism | interfaces vs abstract classes |
| static and default methods |
| Exception handling | Throwable, exception propagation |
| Try-catch |
| Custom exception | throw keyword to explicitly throw an exception |
| throws |
| finally block |
| Creating custom checked and Unchecked Exception |
| try with resource |
| Collections and Generics | ArrayList |
| LinkedList |
| Queue |
| List and Queue methods |
| HashSet and TreeSet |
| Comparator/Comparable |
| Iterator interface |
| Collections and Generics | HashMap and TreeMap |
| Multithreading | Creating threads - extending Thread, implementing Runnable |
| Thread priority |
| Multiple Threads execution |
| Synchronization |
| Multithreading | Executor Service |
| Daemon Thread |
| Functional Interfaces | Functional Interfaces |
| Lambda Expressions | Lambda expressions, Function,Predicate,Consumer, Producer functional interfaces TDD |
| Streams API | FindStream |
| MapStream |
| FlatMapStream |
| MaxStream,MinStream |
| SortedStream |
| 2 | OOAD | SOLID principles | Single Responsibility Principle (SRP) |
| Open-Closed Principle (OCP) |
| SOLID principles | Liskov Substitution Principle (LSP) |
| SOLID principles and Agile Scrum | Interface Segregation Principle (ISP) |
| Dependency Inversion Principle (DIP) |
| Junit | Unit Testing with Junit |
| Assertion based Testing |
| TicTacToe with all test cases | TicTacToe with all test cases |
| Implementing GuitarApp ,OOAD relationships | Relationship |
| Dependency |
| Generalization |
| Association and Aggregation |
| Implementing GuitarApp ,OOAD relationships | Entity set Mapping |
| 3 | Design Patterns | Singleton Pattern | Discuss with example |
| Factory Method/Abstract Factory | Discuss with example |
| Façade Pattern | Discuss with example |
| Decorator Pattern | Discuss with example |
| Strategy Pattern | Discuss with example |
| Observer Pattern | Discuss with example |
| State Pattern | Discuss with example |
| Command Pattern | Discuss with example |
| 4 | Javascript | JavaScript Introduction | Internal and external scripts, Basic syntax |
| JavaScript Arrays | Array functions |
| DOM | Access and manipulating HTML |
| Getting and setting attributes | Getting and setting Html tag attribute, adding and removing from classList |
| JavaScript events | Event Listener, Event Bubbling |
| Time and Date | DateFns |
| Asynchronous JavaScript | Call-back functions, XHR request |
| Promises and fetch API | Promise and Fetch API |
| Worker thread | Use of worker threads |
| JavaScript Patterns | Constructor, Module Pattern |
| 5 | RDBMS /JDBC | RDBMS queries and Normalization | Normalizing sql tables |
| Grouping data |
| Joins |
| Subqueries |
| Views |
| Function and Stored Procedures |
| JDBC | JDBC Architecture |
| Mysql Integration |
| Transactions |
| ResultSet,Connection |
| 6 | JSP /Servlets | Servlet | ServletRequest interface |
| Service doGet and doPost |
| RequestDispatcher |
| sendRedirect |
| ServletContext interface |
| Servlet | Session |
| Cookies |
| Event and Listeners |
| Filters |
| Authentication |
| JSP | Webcontainer and tomcat |
| JSP page Lifecyle |
| JSP Objects - Request, Response, Config, Application, Session, PageContext, Page, Exception |
| Session |
| Application cookie |
| Redirect |
| JSP | JSTL tags |
| CRUD using jsp and servlets |
| 7 | Spring Core | Spring Core | Core spring concepts |
| Dependency injection - XML based, Annotation based |
| Component scan |
| Autowiring |
| spring | core spring concept |
| dependency injection |
| component scan |
| 8 | Spring MVC | MVC | model |
| hybernate validator |
| spring mvc form handling |
| MVC | spring mvc extension handling |
| 9 | Spring Boot & Hibernate | Creating spring boot applications | Creating Spring boot projects |
| Project component |
| Spring Boot Aspect Oriented Programming |
| JPA | JPA Entity and Entity manager |
| Hibernate | Hibernate Mapping |
| Database integration in spring boot | Spring Boot JPA |
| Spring Boot JDBC |
| CRUD operations |
| Database integration in spring boot | Using Entity Manager |
| Creating RESTful APIs | Using RestTemplate and spring CrudRepository |
| Autoconfigurations and Dispatcher Servlet |
| JPA | Collection Mapping |
| Cascading |
| JPQL |
| Postman | Creating a CRUD application and testing |
| JPA | Inheritance |
| Hibernate | Lazy loading |
| Creating RESTful APIs | Internationalization of Restful Services |
| Swagger integration | Configuring auto generation of Swagger |
| Custom Annotation in swagger |
| JWT token based auth concepts | Introduction to JSON Web Tokens |
| JWT token based auth concepts | Authentication |
| Authorization (401 ,403 status code) |
| 10 | Human Engineering | LEARNING | DISCUSSION ON HBR - NETFLIX |
| LEARNING - BABIES, MEANING, SIGNIFICANCE, EXPERIENCE |
| LEARNING - CHALLANGES |
| ATTENTION |
| ACTIVE INVOLVEMENT |
| ERROR FEEDBACK |
| CONSOLIDATION |
| TALENT | TALENT - PROGRAMMING |
| MASTER COACHING |
| JOYFUL | ENERGY |
| CURIOSITY |
| FREEDOM |
| TALENT | DEEP PRACTICE |
| CONSTANT IGNITION |
| JOYFUL | ABUNDANCE |
| PLAY |
| HARMONY |
| TRANSCENDANCE |
| CELEBRATION |
| INTRODUCTION | CHALLANGES |
| STUDENT INTRODUCTION - HOBBIES - GOPAL PATWA |
| LEARNING - DEFINITION, HBR VIDEOS |
| COURSE OUTLINE - LEARNING, AWARENESS, TALENT, JOY |
| AWARENESS | WHO AM I |
| COPING MECHANISMS |
| CONGRUENCE |
| COMPONENTS OF ICEBERG - BEHAVIOR, THINKING, BELIEF, FEELING, LIFE EXP, FEAR, NEED |
| 11 | React | Basics of React Js | Setting up React Environment |
|  | First React Application |
|  | Function Component |
| Functions and Maps | Passing Method Ref and Handling Event |
| Two way databinding |
| working with Conditions |
| using map to rendiner list of items |
| Styles in React | working with styles in react |
| components life cycle events |
| error boundary component |
| Form Validation | handling form elements and their data |
| performing form validations |
| react router |
| invoking server api |
| securing react component |
| Web Hooks | Higher Order components |
| Hooks,userState,useEffect Hook |
| Custom Hooks |
| Fetching data with Hooks |
| 12 | Activities | Project | |
| Activities | Project | |

# Talent Creation Team

**----------------------------------Talent Training Team----------------------------------**

**Mentor**:

1. Yash Shah
2. Swapnil Kulkarni

**Student Coordinator**:

Aditi Talsania

# 

**----------------------------------Project Evaluation Team-------------------------------**

1. Aju Palleri (Sr. Technology Mentor; Swabhav Techlabs),
2. Samruddhi Kotibhaskar (Technology Mentor; Swabhav Techlabs)

# Background of Mentees

Batch comprised 11 individuals who were new to backend development and had no prior experience with full stack java development. Despite their lack of familiarity, these beginners embarked on a learning journey guided by our mentoring program

**No of Mentees**: 11

**Initial Rating of Mentees:**

\*The following ratings are on a scale of 1 to 10 and were determined by the mentor during the initial session, assessing the mentees' skills and knowledge before the course began.

The mentees were assessed on the following parameters:

1. Engineering Knowledge
2. Problem Analysis
3. Design/Dev of Solutions
4. Knowledge of course modules
5. Individual and Teamwork
6. Lifelong learning

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Name of Student** | **Initial Rating** |
| 1 | Sanjay Gangwar | 6.8 |
| 2 | Saurabh Mishra | 7.3 |
| 3 | Sohan Pasi | 6 |
| 4 | swati singh | 4.5 |
| 5 | Ayushi Kumari | 6 |
| 6 | Nitesh Kumar | 6.4 |
| 7 | DharamVeer Verma | 5 |
| 8 | Ramya Pinapothu | 4 |
| 9 | Koppolu Jaideep | 6.1 |
| 10 | Neelu Sharma | 4.5 |
| 11 | Neha y | 5.3 |

# Continuous Evaluation of Mentees

Continuous evaluation of all mentees was carried out during the entire course by the mentor, teaching assistant and student coordinator.

Continuous evaluation was based on 2 primary modes:

1. Behaviour during the session
2. Assignments

## Behaviour during the session

Behaviour of the student was measured by the Teaching Assistants and the mentor on the following parameters during each session:

1. Problem Solving & Debugging
2. Conceptual Understanding
3. Determination and Persistence
4. Speed
5. Solution Optimization & Clean code
6. Attitude & Punctuality
7. Class Engagement

The results of continuous evaluation are as follows:

| **Mentee Name** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **Average Score** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Sanjay Gangwar | 8 | 9 | 8 | 9 | 8 | 8 | 9 | 8.4 |
| Saurabh Mishra | 8 | 8 | 8 | 9 | 9 | 8 | 8 | 8.3 |
| Sohan Pasi | 8 | 8 | 8 | 8 | 8 | 9 | 9 | 8.3 |
| Swati Singh | 8 | 8 | 8 | 8 | 8 | 8 | 9 | 8.2 |
| Ayushi Kumari | 8 | 8 | 8 | 8 | 8 | 9 | 8 | 8.2 |
| Nitesh Kumar | 8 | 8 | 7 | 8 | 8 | 8 | 8 | 7.9 |
| DharamVeer Verma | 8 | 7 | 7 | 7 | 7 | 7 | 7 | 7.1 |
| Ramya Pinapothu | 7 | 7 | 8 | 7 | 7 | 8 | 8 | 7.4 |
| Koppolu Jaideep | 7 | 8 | 8 | 7 | 7 | 8 | 8 | 7.6 |
| Neelu Sharma | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7.0 |
| Neha y | 7 | 6 | 7 | 6 | 7 | 7 | 7 | 6.7 |

## Assignments

All mentees were assigned a total of 20 assignments with varying levels of complexity, ranging from easy to challenging. These assignments encompassed a wide array of programming concepts, such as:

*Data Types in Java, Type Casting, Variables, String, Variable Scope, Function, Jump statement, Variable Scope, Class and Object Digrams, Conditional Statement , Loops, Passing Values, Pass by Reference, Arrays, Maps, Streams, Agile, Abstraction, Objects, Polymorphism, Methods, Error handling, Abstraction, Polymorphism, Exception handling, File I/O, TDD, Database Querying, ACID , Normalization, JOINS, etc.*

Teaching Assistants evaluated and provided feedback for each student's assignments during the session, while the Mentor oversaw the process, guiding the mentees towards improvement.

**No of Assignments**: 4 Easy; 12 Medium; 4 Hard: - Total 20

**Assignment Score**:

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Name of Student** | **Assignment Scores** |
| 1 | Sanjay Gangwar | 9 |
| 2 | Saurabh Mishra | 8 |
| 3 | Sohan Pasi | 8 |
| 4 | swati singh | 9 |
| 5 | Ayushi Kumari | 8 |
| 6 | Nitesh Kumar | 8 |
| 7 | DharamVeer Verma | 8 |
| 8 | Ramya Pinapothu | 8 |
| 9 | Koppolu Jaideep | 8 |
| 10 | Neelu Sharma | 7 |
| 11 | Neha y | 7 |

# Project Evaluation

The project review panel conducted the assessment of the projects.

Mentees were individually assigned one minor project called as ‘Basic Banking Application” in which scope of project is limited and they explored MVC using Servlets and JSP.

They have assigned one major project in a team of two, called “Insurance Application” which has a bigger scope where they implement REST concepts in Spring Boot.

1. **Project Details**
2. **Insurance Application**

Following are the details of project assigned to the teams:

Adopt a well-organized project structure based on the repository pattern to ensure scalability, maintainability, and a clean separation of concerns. This comprehensive setup will provide a secure and efficient insurance app with considerations for various constraints and functionalities.

**Admin module:**

This module is for administrator and employees who work for insurance company. Administrator is the main user of this website. Administrator or Employee can view registered customer and agent records. The administrator handles all types of settings module. Administrator can add different insurance plan details, agent commission details, etc. Even administrator can accept or reject withdrawal request sent by customer and agent.

* Manage Tax and Insurance setting
* Add Employee and Agent
* Insurance type master
* Insurance plan master
* Commission settings
* Withdrawal approval
* Customer report
* Agent report
* Agent wise commission report
* Policy Payment report
* Withdrawal report
* Commission withdrawal report
* Insurance account report
* Transaction report

**Employee module:**

The employee is added by the administrator. Employee can verify the documents submitted by the customer. He can register an agent to the company. He cannot add another employee or make changes in policies.

* Agent Registration
* Manage Profile
* Edit Agent details
* View Commission reports

**Customer module:**

In this module the customer can update their profile details and password. The customer can view registered policy details, track next payment date, number of premium, etc. Customer can withdraw or cancel their registered policy by providing bank account details.

* Profile
* Change password
* Policy account creation
* Policy Claim
* Policy account registration module
* Policy plan details
* Payment Receipt module
* Contact form
* View customer query

The customers or policy holders can register their account and they can create their new policy account. The customer needs to enter their profile details to apply for policy. The system displays policy information and plan details before applying policy. After policy registration the system generates policy receipt. The customers can send queries using contact form. Employees can view customers query in the customer query panel.

**Insurance Agent module:**

The agents work for insurance company who provides information regarding the policies and schemes and brings new customers to the insurance company. The agents earn commission for each and every policy registration. Admin adds agent by verifying their profile manually.

* Agent Registration module
* Agent Login module
* Agent profile module
* Change Password module
* Policy registration module
* Agent commission module
* Commission report module
* Earnings report
* Withdraw commission module

***The concepts used in the backend REST development using Springboot***

1. Create controller, service, and repository layers.
2. Exception handling
3. Adding pagination and sorting
4. File upload and download
5. Email to customers
6. Entity mapping
7. Spring security using JWT Authentication
8. Logging

***The concepts used in the frontend development using Angular***

1. Create angular component
2. Creating modules
3. Reactive Forms with validations
4. Routing
5. Services
6. Pagination
7. Authentication and interceptors
8. Bootstrap or Material UI

## **Project Assessment**

The project review panel undertook the evaluation of the projects, focusing on individual contributions. The assessment included a project demo, coupled with questions centred around the topics covered during the course.

Mentees were assessed on the following parameters:

1. Engineering Knowledge
2. Problem Analysis
3. Design/Dev of Solutions
4. Knowledge of course modules
5. Individual and Teamwork
6. Lifelong learning

All mentees finished “Basic Banking Application” using Servlets and JSP in 3 to 4 days and expected to finish “Advanced Banking Application” using REST in about 9 to 10 days.

The assessment results for the projects were as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Team No** | **Name** | **Banking Application (Servlet/JSP)** | **Insurance Application (Spring Boot)** |
| 1 | Ayushi Kumari | 8.1 | 8.2 |
| Sohan Pasi | 8.2 | 8.3 |
| 2 | Sanjay Gangwar | 8.3 | 8.5 |
| Koppolu Jaideep | 7.2 | 7.6 |
| 3 | Swati Singh | 8.4 | 8.2 |
| Nitesh Kumar | 7.8 | 7.9 |
| 4 | Saurabh Mishra | 8.2 | 8.3 |
| Neelu Sharma | 7.7 | 6.8 |
| 5 | Ramya Pinapothu | 7.0 | 6.9 |
| Dharamveer Verma | 6.0 | 7.1 |
| Neha Y. | 7.1 | 6.7 |

# Growth Report of Batch

Evaluation of the growth / progress of mentees is based on

1. Continuous Evaluation (refer section 4)
2. Project Evaluation (refer section 5)

## Continuous Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.No.** | **Name of Student** | **Initial Rating** | **Cont. Evaluation** | **Growth** |
| 1 | Sanjay Gangwar | 6.8 | 8.4 | 1.6 |
| 2 | Saurabh Mishra | 7.3 | 8.3 | 1 |
| 3 | Sohan Pasi | 6 | 8.3 | 2.3 |
| 4 | swati singh | 4.5 | 8.2 | 3.7 |
| 5 | Ayushi Kumari | 6 | 8.2 | 2.2 |
| 6 | Nitesh Kumar | 6.4 | 7.9 | 1.5 |
| 7 | DharamVeer Verma | 5 | 7.1 | 2.1 |
| 8 | Ramya Pinapothu | 4 | 7.4 | 3.4 |
| 9 | Koppolu Jaideep | 6.1 | 7.6 | 1.5 |
| 10 | Neelu Sharma | 4.5 | 7 | 2.5 |
| 11 | Neha y | 5.3 | 6.7 | 1.4 |

## Project Evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.No.** | **Name of Student** | **Initial Rating** | **Project Demo** | **Growth** |
| 1 | Sanjay Gangwar | 6.8 | 8.5 | 1.7 |
| 2 | Saurabh Mishra | 7.3 | 8.3 | 1 |
| 3 | Sohan Pasi | 6 | 8.2 | 2.2 |
| 4 | swati singh | 4.5 | 8.2 | 3.7 |
| 5 | Ayushi Kumari | 6 | 8.2 | 2.2 |
| 6 | Nitesh Kumar | 6.4 | 7.9 | 1.5 |
| 7 | DharamVeer Verma | 5 | 7.1 | 2.1 |
| 8 | Ramya Pinapothu | 4 | 6.9 | 2.9 |
| 9 | Koppolu Jaideep | 6.1 | 7.6 | 1.5 |
| 10 | Neelu Sharma | 4.5 | 6.8 | 2.3 |
| 11 | Neha y | 5.3 | 6.7 | 1.4 |

## Overall Growth Analysis of the Batch:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Initial Rating | Average Rating (Continuous Evaluation) | Growth (As per continuous evaluation) | Post Batch Rating (After Project Demo) | Growth (As per Project evaluation) |
| Overall Batch | **5.62** | **7.73** | **2.11** | **7.62** | **2.0** |

**Overall batch has grown more than 2.0 points**

## Top Performers

**Sanjay Gangwar, Ayushi Kumari, Sohan Pasi and Swati Singh** have demonstrated exceptional performance and exhibit the potential to become exemplary leaders within Monocept.

A special mention to **Saurabh Mishra, Nitesh Kumar and Koppolu Jaideep** who have shown remarkable self-learning capabilities. They have the potential to be valuable assets for Monocept in the future, contributing significantly to the company's growth.

**Ramya Pinapothu** and **Neha y** are hardworking and sincere and shown good progress despite having week basics.

**DharamVeer Verma** and **Neelu Sharma** have also progressed well.

# Feedbacks and Observations

## a. Mentors Feedback on Overall batch:

* The entire batch is demonstrating a highly positive approach when it came to tackling tasks and challenges.
* The mentees made a significant progress during the batch.
* Each mentee exhibits unique strengths. If nurtured well, could become a great resource for the company.
* Mentees now feel more confident in their abilities as a software developer and better prepared for the challenges that lie ahead in their career.

## b. Observations by the Mentor:

* The academic syllabus mentees have undergone does not address the real needs of modern approaches to software development.
* Mentees exhibited a lack of awareness concerning security policies, security algorithms, design principles, and the fundamentals of writing clean code at the start of batch.

## c. Recommendation by the Mentor:

* More focus on normalization and optimization of data would help in terms of overall application functioning
* Insights on deployment process

## d. Mentees Feedback about the Course:

* The mentees are sharing positive feedbacks about the mentorship program, mentioning that they appreciated the personalized guidance and the opportunity to work on a practical project.
* Mentors are consistently getting rated as highly knowledgeable and engaging, with scores ranging from 9 to 10
* Mentees are giving highly positive feedback on curriculum, mentors overall support, projects assigned and total learning experience.
* Overall, mentees are expressing high satisfaction with the training, with satisfaction ratings ranging from 9 to 10.