SPEC DRIVEN DEVELOPMENT

An AI-Native Software Development Program

Beyond vibe-coding, building reliable systems.

Students learn to specify, build, test and defend real applications with AI coding agents - then prove their skills in a team hackathon.

36 HOURS
TOTAL

24 HOURS
LEARNING + PRACTICE

12 HOURS
REAL-WORLD HACKATHON



THE MOMENT

The gap isn’t code generation. It’s engineering confidence.

AI can produce an application quickly. Employers still need people who can explain what should happen, why it should happen, and how to know whether it works.

66%

of survey respondents use AI coding assistants or agents

Stack Overflow Developer Survey 2026 [1]

48%

trust AI output when they can easily verify it

Stack Overflow Developer Survey 2026 [1]

WHAT THE INDUSTRY NEEDS

People who can translate intent into clear requirements, supervise AI-generated implementation, review changes and verify outcomes. SDD, BDD and TDD make these practices teachable.

[1] Stack Overflow Developer Survey 2026

[2] DORA 2025 industry research

THE IDEA

Three disciplines.
One dependable workflow.

Not another collection of buzzwords. Each discipline answers a simple engineering question.

01

SDD

WHAT must be built?

Agree on requirements, constraints and a reviewable implementation plan.

02

BDD

HOW should it behave?

Describe user scenarios in precise Given / When / Then terms.

03

TDD

HOW do we verify it?

Test first, implement the smallest change and refactor safely.

REQUIREMENT

Registration count must never exceed event capacity.

BEHAVIOR

Given a full event, when someone registers, reject the request.

TEST

Assert that confirmed registrations never exceed capacity.

THE LEARNING JOURNEY

Six days. One transformation.

DAYS 01-04 / LEARN + PRACTICE

01 UNDERSTAND

AI realities, SDD / BDD / TDD, problem discovery

Output: Problem brief + acceptance criteria

02 SPECIFY

Spec Kit, EARS requirements, context, planning

Output: Reviewed specification + task plan

03 BUILD + TEST

Coding agents, BDD, TDD, debugging

Output: Working feature + test evidence

04 ADAPT

Existing codebase, regression, safe changes

Output: Updated spec + verified fix

DAYS 05-06 / HACKATHON

05 CREATE

Select problem, approve specification, plan and build core MVP

Output: Approved spec + functional core workflow

06 DELIVER

Integrate, handle surprise change, test and present

Output: Verified app + evidence-backed demo

INCLUSIVE BY DESIGN

One room. Different starting points. No passive spectators.

One shared project, three levels of challenge, and the same core standards for every learner.

LEVEL 01

FOUNDATION

Starter code + guided labs

Write acceptance criteria, modify a small function and run tests.

LEVEL 02

BUILDER

Independent feature work

Implement features, tests, debugging and Git collaboration.

LEVEL 03

ADVANCED

Stretch engineering tasks

Handle complex edge cases, change impact and stronger test design.

EVERY STUDENT MUST

Write a testable requirement. Make a real code change. Run and interpret a test. Explain the result. Participate in review.

EIGHT PRACTICAL LABS • ONE SHARED PROJECT ON DAYS 1-4 • INDIVIDUAL PROFICIENCY CHECKS

THE CAPSTONE

The hackathon is not a demo contest.

It is a 12-hour simulation of how reliable software gets delivered: scope the problem, approve the specification, build in reviewable slices, test behavior, and defend decisions.

01
DEFINE
Real users + bounded MVP

›

02
APPROVE
Mentor-reviewed specification

›

03
BUILD
AI-guided implementation

›

04
VERIFY
Tests + regression evidence

›

05
DEFEND
Working demo + team Q&A;

THE REAL-WORLD TWIST

Change request: “The event is full. Add a waiting list without breaking registrations.”

Update spec → revise tests → implement safely

EVIDENCE, NOT JUST APPEARANCE

A working user journey, versioned specification, readable source code, test results, change record, and a clear explanation of limitations.

MEASURED OUTCOMES

What leaves the room on Day 6?

Not only a certificate: a portfolio of decisions, tests and working software that learners can demonstrate and explain.

<> spec.md

PROBLEM + ACCEPTANCE CRITERIA

<> plan.md

DESIGN + TASK BREAKDOWN

<> app/

WORKING SOFTWARE

<> tests/

VERIFICATION EVIDENCE

<> changes.md

REQUIREMENTS CHANGE

<> demo

LIVE TECHNICAL WALKTHROUGH

INSTITUTIONAL OUTPUTS

BASELINE + EXIT
Comparable skills check

PROJECT REVIEW
Evidence-driven evaluation

COHORT SUMMARY
Consolidated outcomes

CAREER RELEVANCE

Skills students can take into interviews.

The program supports established entry-level pathways by giving students artifacts they can explain. It does not claim to create a new “SDD engineer” job market or guarantee placement.

01

SOFTWARE ENGINEERING

Associate Developer · Junior Software Engineer

Requirements → implementation tasks → code review → working application

Interview evidence: Demonstrate a feature, a Git commit and the acceptance criteria it satisfies.

02

QUALITY + AUTOMATION

QA Engineer · Junior SDET · Test Automation

BDD scenarios → tests → negative cases → defect fixes

Interview evidence: Show a failing test, the defect, the correction and regression evidence.

03

PRODUCT + REQUIREMENTS

Product Intern · Junior Business Analyst · Implementation Associate

User stories → requirement clarity → scope → change control

Interview evidence: Defend a specification, clarify assumptions and justify a change decision.

These roles require additional programming, reasoning, tool and employer-specific qualifications.

DELIVERY CREDIBILITY

A builder who teaches people to build.



SAI PAVAN VELIDANDLA
Founder & CEO, RedRuff
Technology educator, builder and mentor

LEAD FACILITATOR

17+ years in technology training

His supplied professional portfolio describes work spanning AI, software development, faculty development, research, hands-on training and student mentorship.

MEET THE FACILITATOR / pavan.redruff.in

2026 / GEETHANJALI VIEW SOURCE

Six-day MERN program; 108 second-year CSE students and about 40 projects reported in the portfolio.

2024 / ANITS VIEW SOURCE

Student participants credit Pavan with mentoring an award-winning AR science lab hackathon team.

2024 / LBRCE VIEW SOURCE

An official college report names Pavan as a resource person in a six-day blockchain workshop.

“I don’t just teach technology. I help people build.”

REDRUFF • LEARNING LAB

LET'S BUILD THE NEXT GENERATION OF PROBLEM SOLVERS.

Bring AI-native software engineering to your campus.

Partner with RedRuff on a six-day program designed for demonstrable student capability, working applications and visible outcomes.

REDRUFF PROVIDES

- Lead instructor and agreed technical mentors
- Guided labs, starter repositories and templates
- Hackathon challenges, review and assessment
- Project showcase and cohort outcomes summary

THE INSTITUTION PROVIDES

- Student nominations and faculty coordination
- Computer lab, internet and basic infrastructure
- Scheduling and on-campus support
- Faculty participation in reviews and showcase

EXPLORE THE PROGRAM

sdd.redruff.in

Presenter profile: pavan.redruff.in

pavan@redruff.in | +91 95024 44362

Cohort size, dates, staffing and commercial terms to be agreed with the institution.

SCAN TO EXPLORE