**SPM TOOLBOX**

**Objective**

This project aims to create a Flutter-based mobile app to support students and professionals in mastering Software Project Management (SPM) concepts through practical tools :

* Cost Benefit Analysis
* Process Model Selection
* PERT Estimation
* Critical Path Calculation

The app bridges theoretical SPM knowledge with hands-on application.

**Code Snippets for Key Functions**

1. **Cost Benefit Analysis**

**CostBenefitResult calculateCBA(double initial, List<double> flows, double rate, int lifespan) {**

**double npv = -initial + flows.asMap().entries.map((e) => e.value / pow(1 + rate, e.key + 1)).reduce((a, b) => a + b);**

**double netProfit = flows.reduce((a, b) => a + b) - initial;**

**double roi = netProfit / initial \* 100;**

**return CostBenefitResult(**

**initialInvestment: initial,**

**cashFlows: flows,**

**discountRate: rate,**

**projectLifespan: lifespan,**

**netProfit: netProfit,**

**averageAnnualProfit: netProfit / lifespan,**

**roi: roi,**

**npv: npv,**

**irr: 0.0, // Simplified; actual IRR requires iterative calculation**

**npvCalculationTable: [],**

**irrComparisonTable: [],**

**);**

**}**

1. **Process Model Selection**

**ProcessModelResult selectProcessModel(String projectType) {**

**if (projectType == "small") {**

**return ProcessModelResult(**

**modelName: "Waterfall",**

**justification: "Simple, predictable projects",**

**characteristics: ["Linear", "Defined stages"],**

**);**

**} else {**

**return ProcessModelResult(**

**modelName: "Agile",**

**justification: "Flexible, iterative projects",**

**characteristics: ["Iterative", "Adaptive"],**

**);**

**}**

**}**

1. **PERT Estimation**

**PertEstimationResult calculatePERT(List<PertActivity> activities) {**

**for (var act in activities) {**

**act.expectedTime = (act.optimisticTime + 4 \* act.mostLikelyTime + act.pessimisticTime) / 6;**

**act.variance = pow((act.pessimisticTime - act.optimisticTime) / 6, 2);**

**}**

**var criticalPath = []; // Logic to determine critical path (simplified here)**

**double totalDuration = activities.map((a) => a.expectedTime!).reduce(max);**

**double totalVariance = activities.map((a) => a.variance!).reduce((a, b) => a + b);**

**return PertEstimationResult(**

**activities: activities,**

**criticalPath: criticalPath,**

**totalProjectDuration: totalDuration,**

**totalVariance: totalVariance,**

**standardDeviation: sqrt(totalVariance),**

**zScore: 0.0, // Requires target time input**

**completionProbability: 0.0, // Simplified**

**);**

**}**

1. **Critical Path Calculation**

**void calculateCriticalPath(List<Activity> activities) {**

**// Forward Pass**

**for (var activity in activities) {**

**activity.earlyStart = activity.predecessors.isEmpty**

**? 0**

**: activities.where((a) => activity.predecessors.contains(a.id)).map((a) => a.earlyFinish).reduce(max);**

**activity.earlyFinish = activity.earlyStart + activity.duration;**

**}**

**// Backward Pass**

**var maxFinish = activities.map((a) => a.earlyFinish).reduce(max);**

**for (var activity in activities.reversed) {**

**activity.lateFinish = activity == activities.last ? maxFinish : activities.where((a) => a.predecessors.contains(activity.id)).map((a) => a.lateStart).reduce(min);**

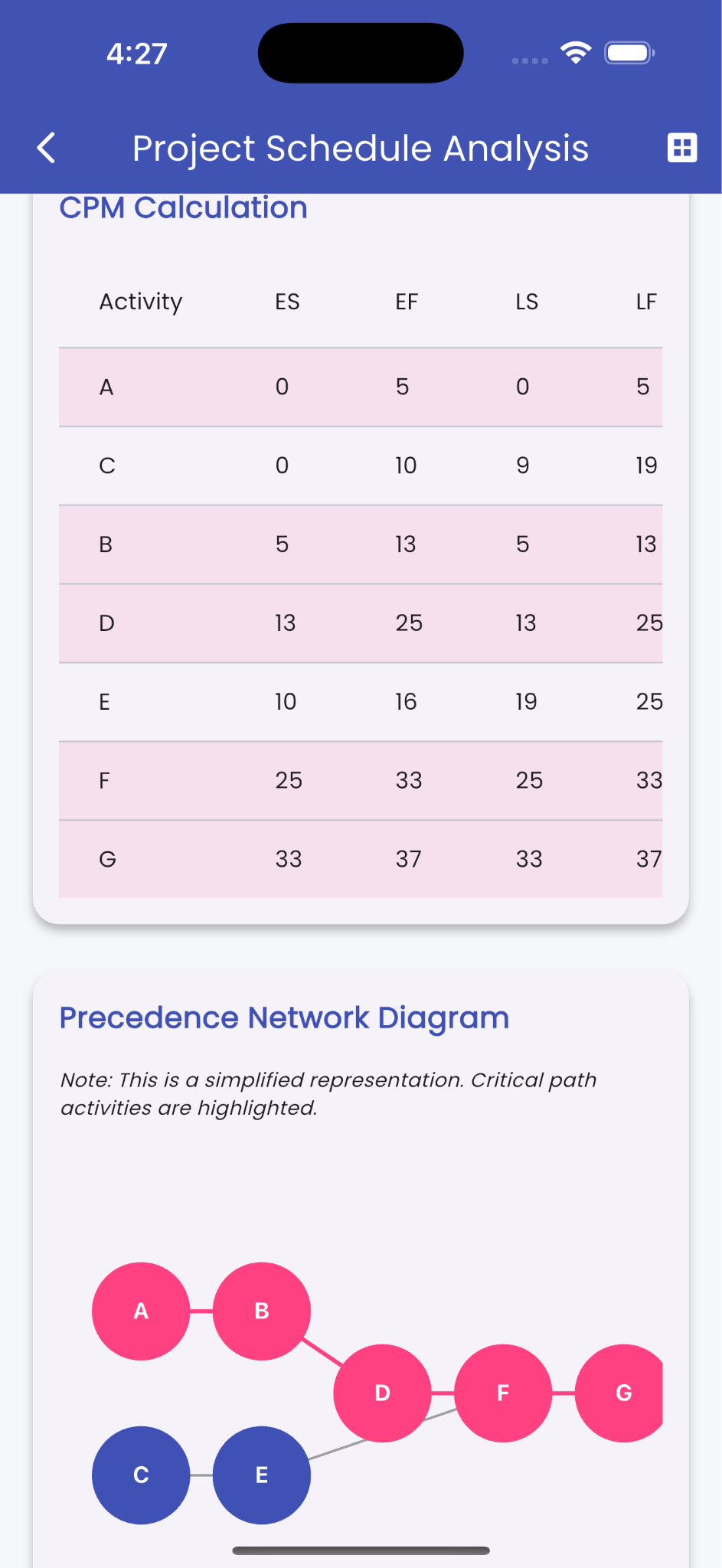
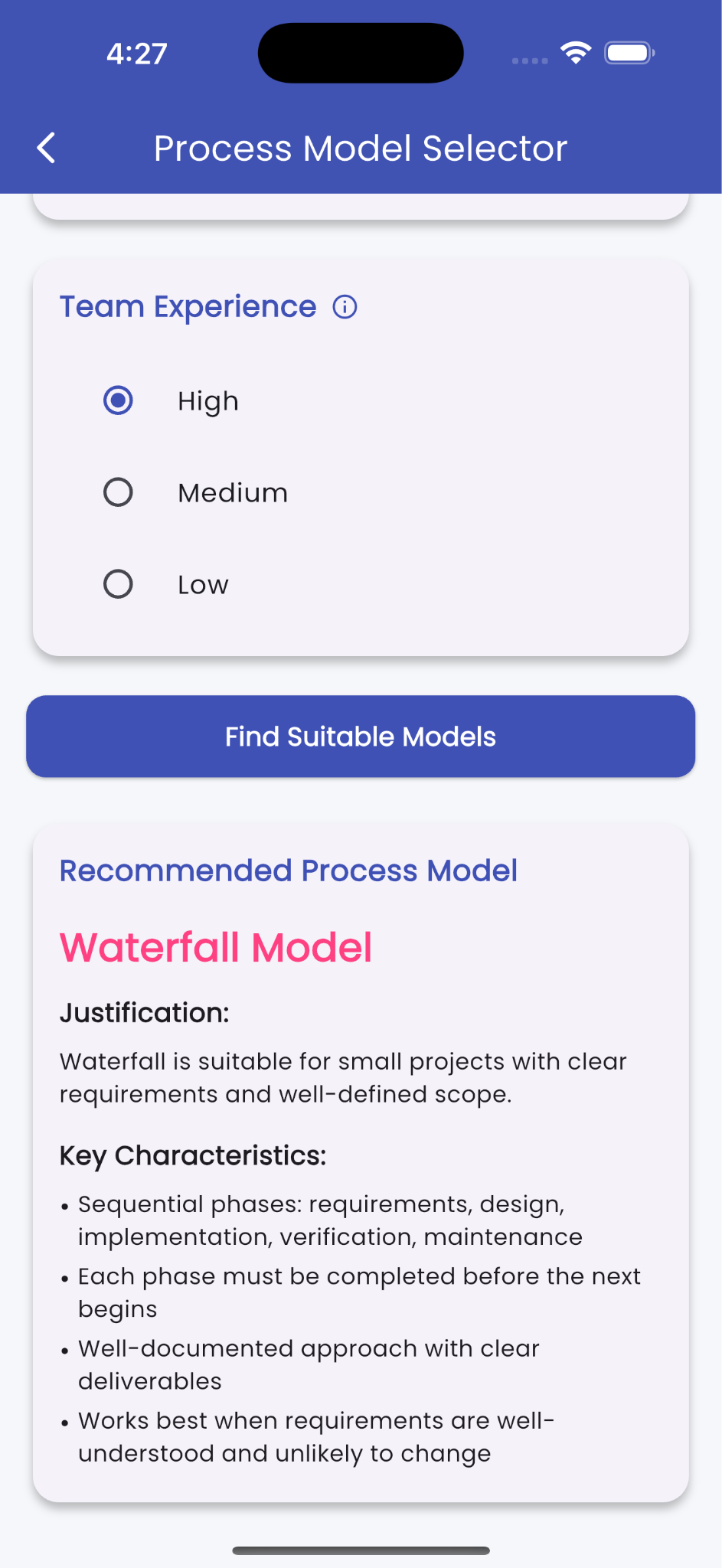
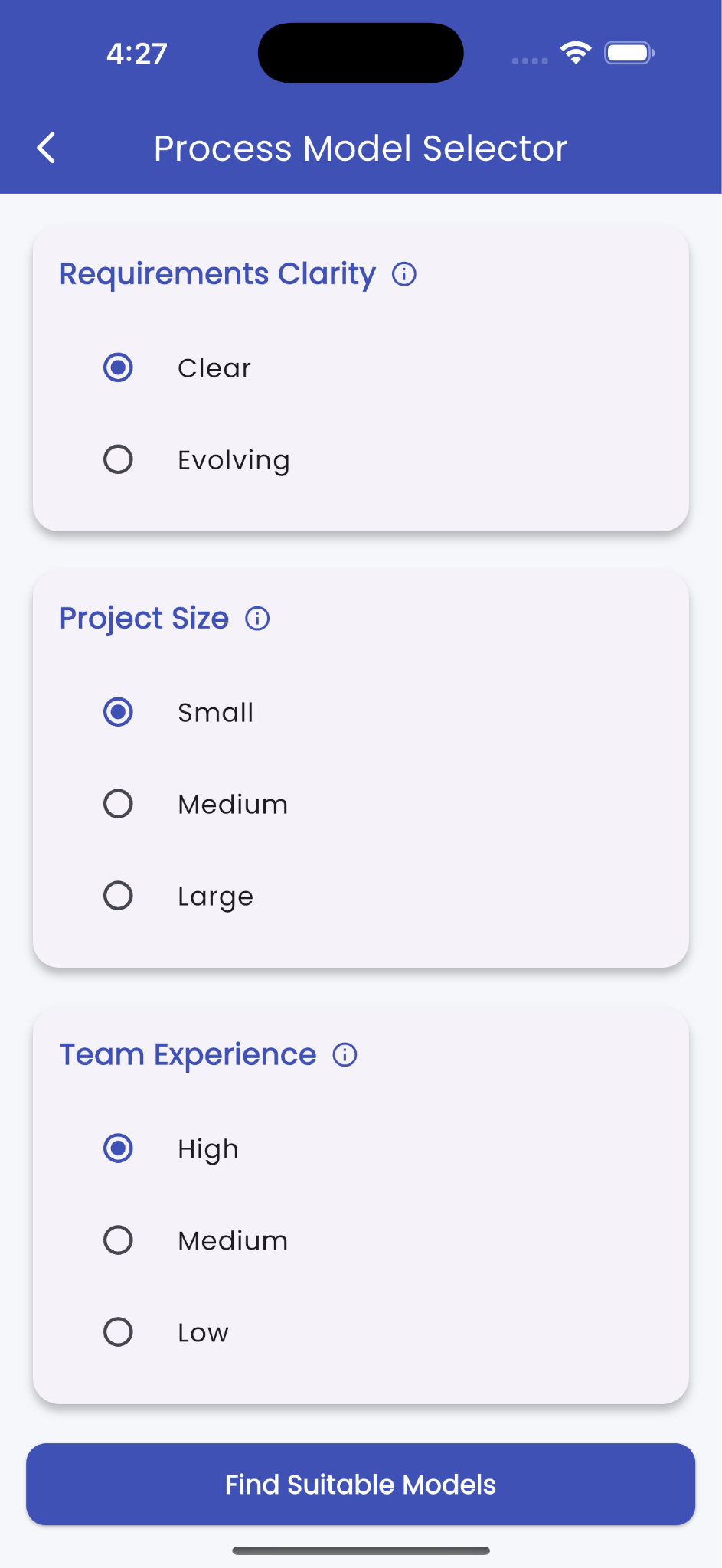
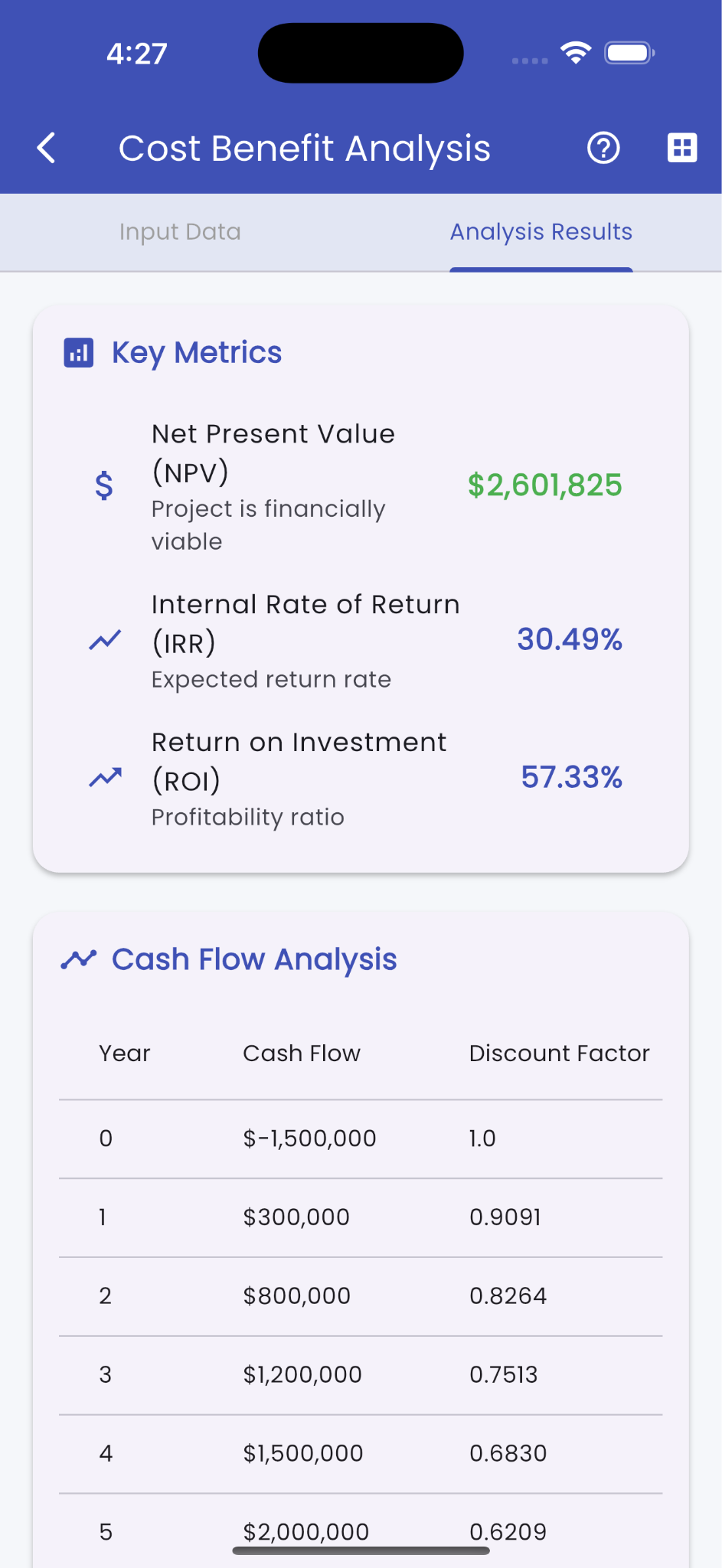
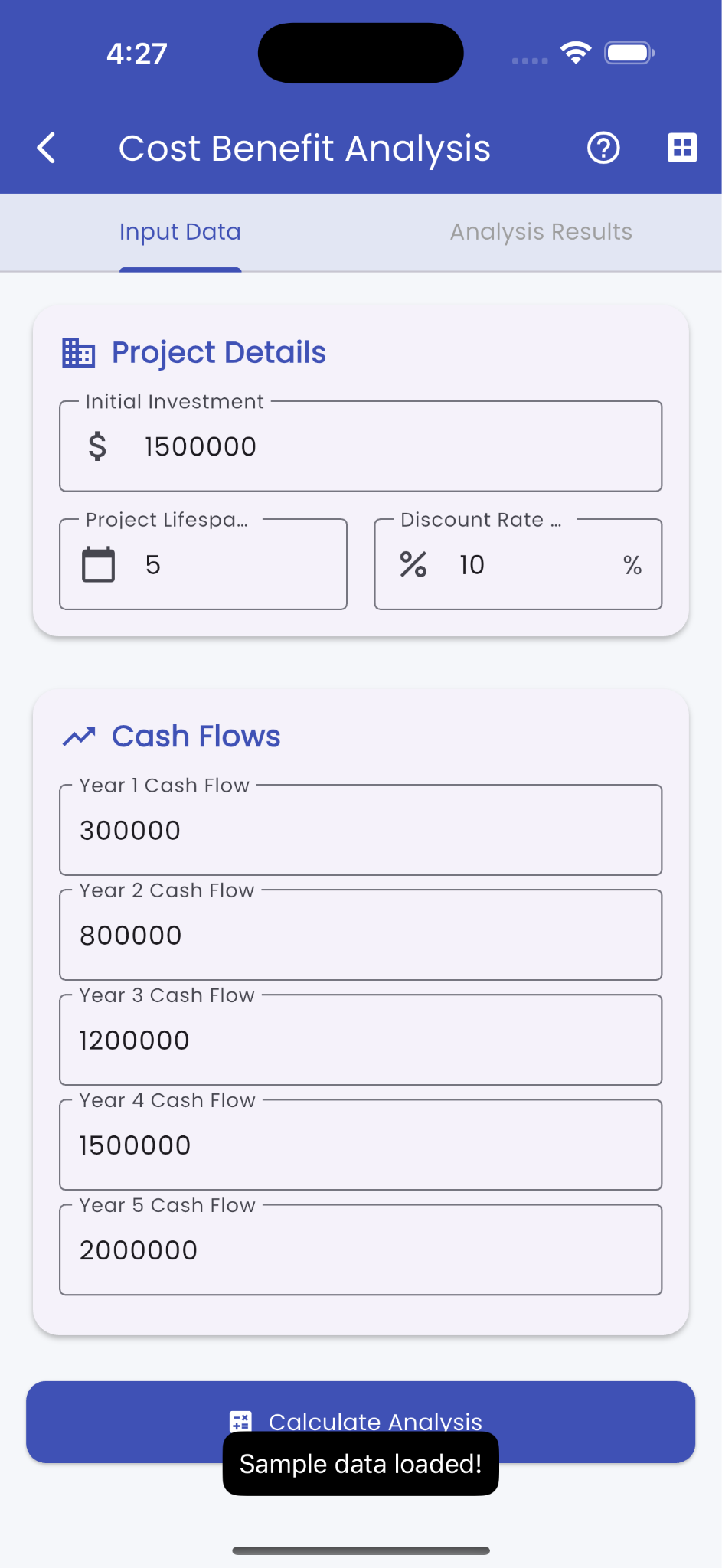
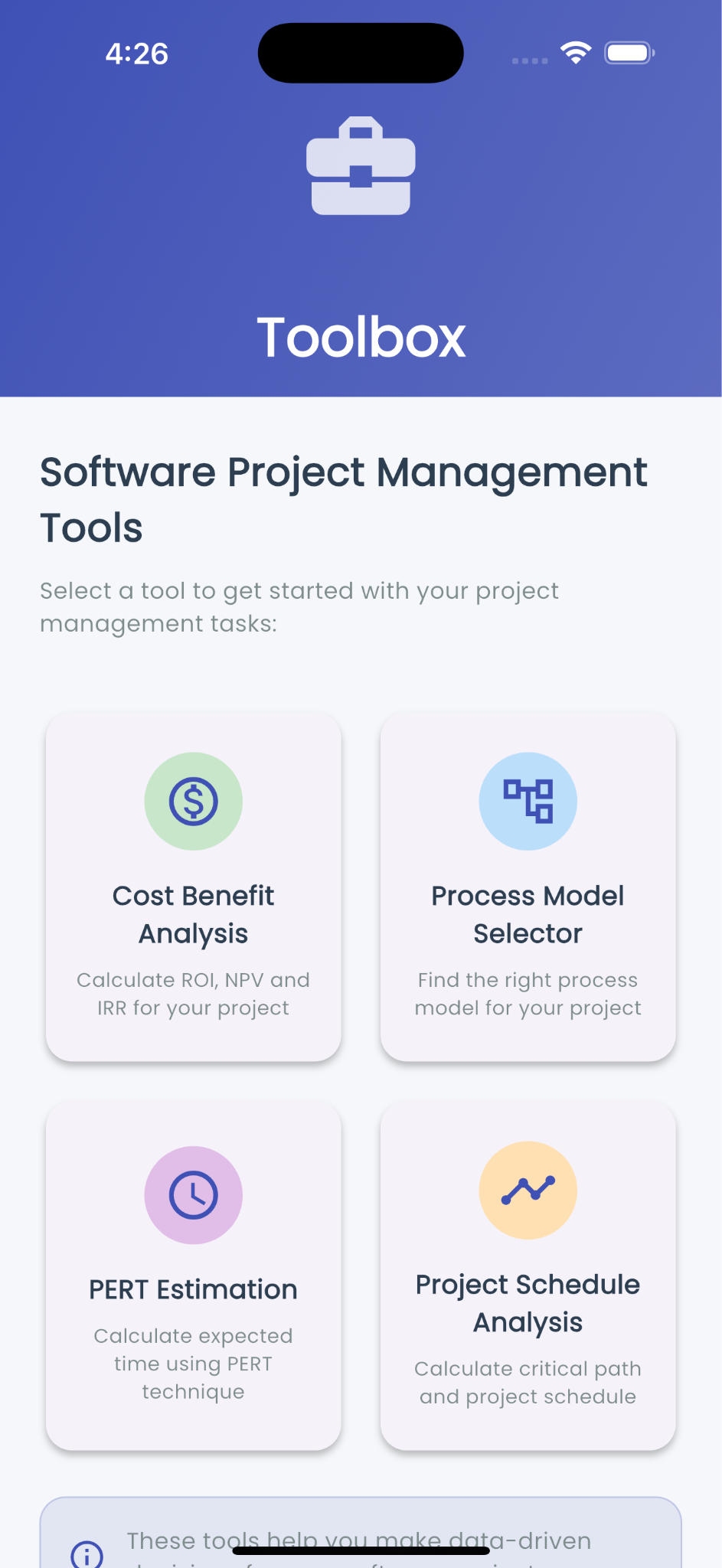
**activity.lateStart = activity.lateFinish - activity.duration;**

**activity.totalFloat = activity.lateStart - activity.earlyStart;**

**activity.isCritical = activity.totalFloat == 0;**

**}**

**}**

****

**Tools / Technologies**

* Flutter (UI toolkit for natively compiled mobile applications)
* Dart (Programming language for Flutter)
* VS code (Development environment)
* Firebase (optional – if used for data storage or analytics)
* Git & GitHub (for version control)

## **Future Scope**

## Include Risk Analysis and Resource Planning tools

* Export results to PDF or Excel.
* Add user accounts for saving project data.
* Integrate cloud database to store and retrieve project history.
* Add AI-based suggestions for model selection or cost estimation.