```
package hello;
import java.util.*;

//---------------- EMPLOYEE CLASS ----------------
class Employee {
 private int empId;
 private String name;
 private String dept;
 private String email;

 // Constructor
 public Employee(int empId, String name, String dept, String email) {
    this.empId = empId;
    this.name = name;
    this.dept = dept;
    this.email = email;
 }

 // Getters & Setters
 public int getEmpId() { return empId; }
 public String getName() { return name; }
 public String getDept() { return dept; }
 public String getEmail() { return email; }

 public void setDept(String dept) { this.dept = dept; }
 public void setEmail(String email) { this.email = email; }

 // ToString
 @Override
 public String toString() {
    return empId + " - " + name + " (" + dept + ")";
 }
}

//---------------- MANAGER CLASS ----------------
class Manager extends Employee {
 private double approvalLimit;

 // Constructor
 public Manager(int empId, String name, String dept, String email, double approvalLimit)
{
    super(empId, name, dept, email);
    this.approvalLimit = approvalLimit;
 }

 // Approve Method (Overridden)
 public boolean approve(Expense expense) {
    if (expense.getAmount() <= approvalLimit) {
       expense.setStatus("APPROVED");
       return true;
    } else {
       expense.setStatus("REJECTED - Over Limit");
       return false;
    }
 }
 public double getApprovalLimit() { return approvalLimit; }
 public void setApprovalLimit(double approvalLimit) { this.approvalLimit = approvalLimit; }
}

//---------------- EXPENSE CLASS ----------------
class Expense {
 private int expenseId;
 private String type; // Travel, Meals, etc.
 private double amount;
 private String status; // SUBMITTED, APPROVED, REJECTED, REIMBURSED
 private Employee requester;

 // Constructor
 public Expense(int expenseId, String type, double amount, Employee requester) {
    this.expenseId = expenseId;
    this.type = type;
    this.amount = amount;
    this.status = "SUBMITTED";
    this.requester = requester;
 }

 // Getters & Setters
 public int getExpenseId() { return expenseId; }
 public String getType() { return type; }
 public double getAmount() { return amount; }
 public String getStatus() { return status; }
 public Employee getRequester() { return requester; }
 protected void setStatus(String status) { this.status = status; }

 @Override
 public String toString() {
   return "ExpenseID: " + expenseId + ", Type: " + type + ", Amount: " + amount +
        ", Status: " + status + ", Requester: " + requester.getName();
 }
}

//---------------- EXPENSE SERVICE ----------------
class ExpenseService {
 private List<Expense> expenseList = new ArrayList<>();
 private int nextExpenseId = 1;

 // Submit Expense (Overloaded)
 public Expense submit(Employee emp, String type, double amount) {
   Expense exp = new Expense(nextExpenseId++, type, amount, emp);
   expenseList.add(exp);
   return exp;
 }

 public Expense submit(Employee emp, String type, double amount, String attachment)
{
   Expense exp = new Expense(nextExpenseId++, type + " (Attachment: " + attachment +
")", amount, emp);
   expenseList.add(exp);
   return exp;
 }
```

```
// Approve Expense
public void approveExpense(Manager mgr, int expenseId) {
  for (Expense e : expenseList) {
    if (e.getExpenseId() == expenseId) {
      mgr.approve(e); // Polymorphism: Manager overrides behavior
    }
  }
}

// Reject Expense
public void rejectExpense(int expenseId) {
  for (Expense e : expenseList) {
    if (e.getExpenseId() == expenseId) {
      e.setStatus("REJECTED");
    }
  }
}

// Mark Reimbursed
public void markReimbursed(int expenseId) {
  for (Expense e : expenseList) {
    if (e.getExpenseId() == expenseId && e.getStatus().equals("APPROVED")) {
      e.setStatus("REIMBURSED");
    }
  }
}
```

```
// List by Status
public void listByStatus(String status) {
    System.out.println("\nExpenses with status: " + status);
    for (Expense e : expenseList) {
        if (e.getStatus().equalsIgnoreCase(status)) {
            System.out.println(e);
        }
    }
}

// Calculate Total per Employee
public double calculateTotal(Employee emp) {
    double total = 0;
    for (Expense e : expenseList) {
        if (e.getRequester().getEmpId() == emp.getEmpId() && e.getStatus().equals("REIMBURSED")) {
            total += e.getAmount();
        }
    }
    return total;
}

// Print Monthly Summary
public void monthlySummary() {
    System.out.println("\n--- Monthly Expense Summary ---");
    Map<String, Double> deptTotal = new HashMap<>();
    for (Expense e : expenseList) {
        deptTotal.put(e.getRequester().getDept(),
            deptTotal.getOrDefault(e.getRequester().getDept(), 0.0) + e.getAmount());
    }
    for (String dept : deptTotal.keySet()) {
       System.out.println("Dept: " + dept + ", Total: " + deptTotal.get(dept));
    }
 }
}

//---------------- MAIN CLASS ----------------
public class eappmain {
 public static void main(String[] args) {
    ExpenseService service = new ExpenseService();

    // Employees and Managers
    Employee emp1 = new Employee(101, "Alice", "IT", "alice@company.com");
    Employee emp2 = new Employee(102, "Bob", "Finance", "bob@company.com");
    Manager mgr1 = new Manager(201, "Charles", "IT", "charles@company.com", 5000);

    // Submit Expenses
    Expense e1 = service.submit(emp1, "Travel", 2000);
    Expense e2 = service.submit(emp1, "Meals", 800, "MealReceipt.pdf");
    Expense e3 = service.submit(emp2, "Travel", 6000);

    // Approvals
    service.approveExpense(mgr1, e1.getExpenseId()); // Approved
    service.approveExpense(mgr1, e2.getExpenseId()); // Approved
    service.approveExpense(mgr1, e3.getExpenseId()); // Rejected (over limit)
```

```
        // Mark reimbursed
        service.markReimbursed(e1.getExpenseId());
        service.markReimbursed(e2.getExpenseId());

        // Reports
        service.listByStatus("APPROVED");
        service.listByStatus("REJECTED");
        service.listByStatus("REIMBURSED");

        System.out.println("\nTotal reimbursed for Alice: " + service.calculateTotal(emp1));
        System.out.println("Total reimbursed for Bob: " + service.calculateTotal(emp2));

        service.monthlySummary();
    }
}
```