CHAPTER 4

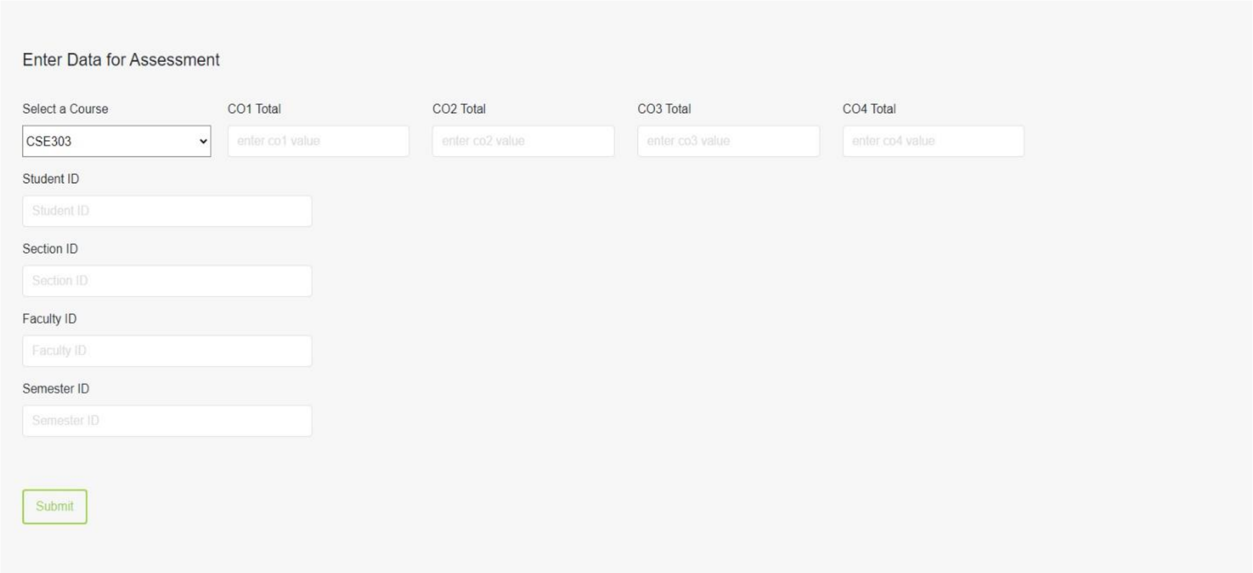
PHYSICAL SYSTEM DESIGN

INPUT FORMS

OUTPUT FORMS

INPUT FORMS

Enrollment Data Entry for Assessment



Enter Data for Assessment

Select a Course: CSE303

CO1 Total: enter co1 value

CO2 Total: enter co2 value

CO3 Total: enter co3 value

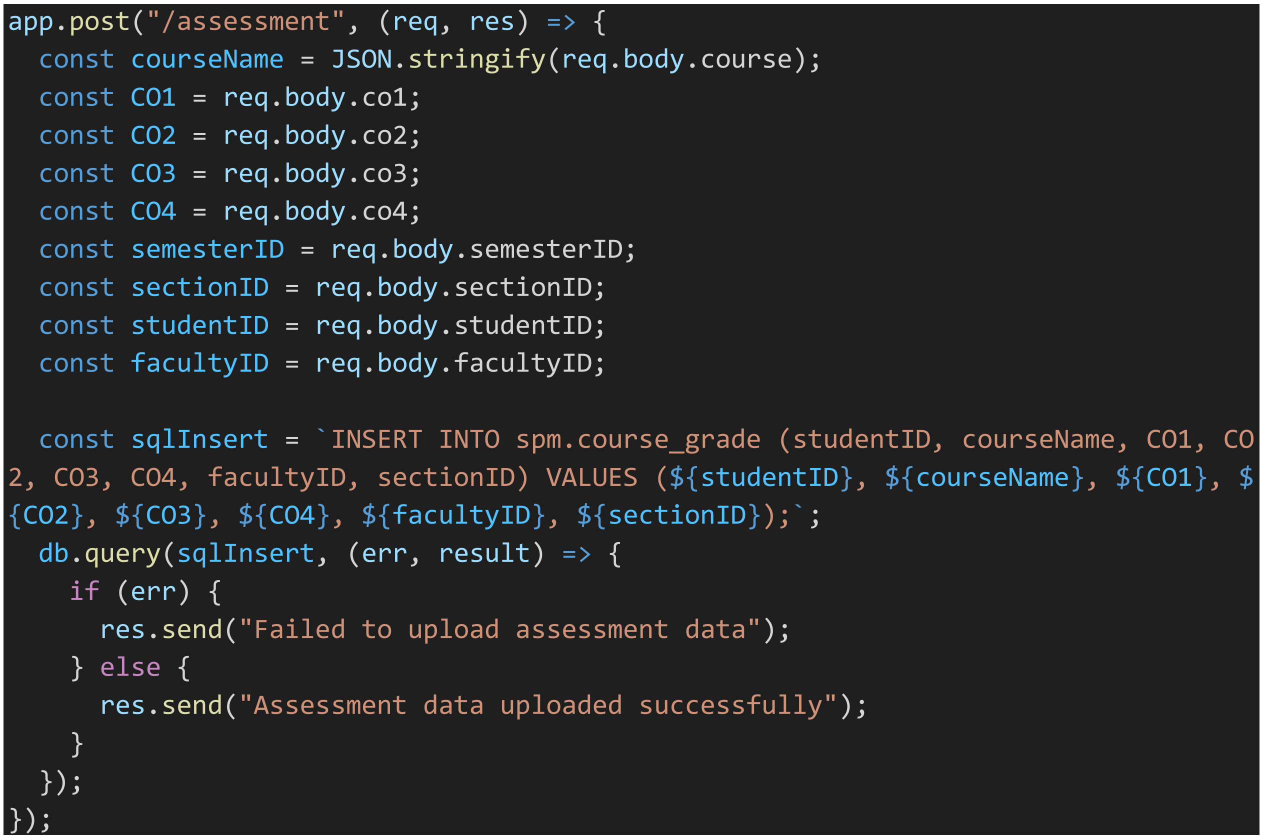
CO4 Total: enter co4 value

Student ID: Student ID

Section ID: Section ID

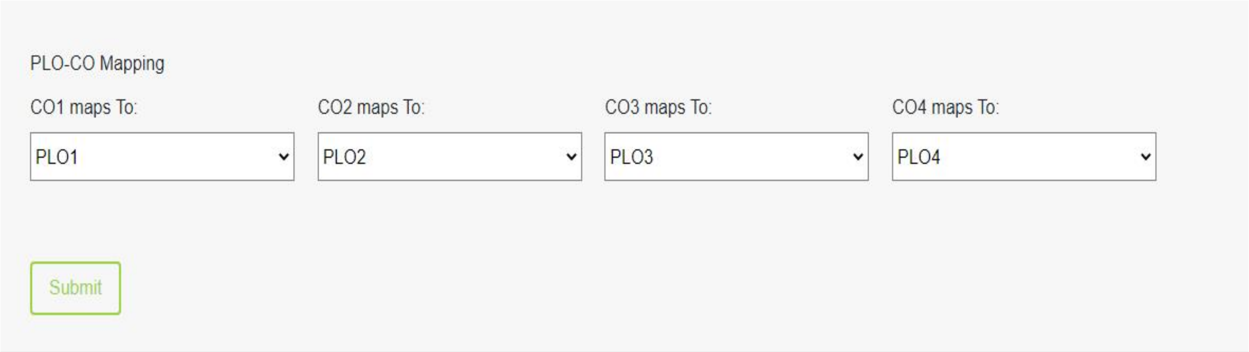
Faculty ID: Faculty ID

Semester ID: Semester ID



```
app.post("/assessment", (req, res) => {
  const courseName = JSON.stringify(req.body.course);
  const CO1 = req.body.co1;
  const CO2 = req.body.co2;
  const CO3 = req.body.co3;
  const CO4 = req.body.co4;
  const semesterID = req.body.semesterID;
  const sectionID = req.body.sectionID;
  const studentID = req.body.studentID;
  const facultyID = req.body.facultyID;

  const sqlInsert = `INSERT INTO spm.course_grade (studentID, courseName, CO1, CO2, CO3, CO4, facultyID, sectionID) VALUES (${studentID}, ${courseName}, ${CO1}, ${CO2}, ${CO3}, ${CO4}, ${facultyID}, ${sectionID});`;
  db.query(sqlInsert, (err, result) => {
    if (err) {
      res.send("Failed to upload assessment data");
    } else {
      res.send("Assessment data uploaded successfully");
    }
  });
});
```

PLO-CO Mapping



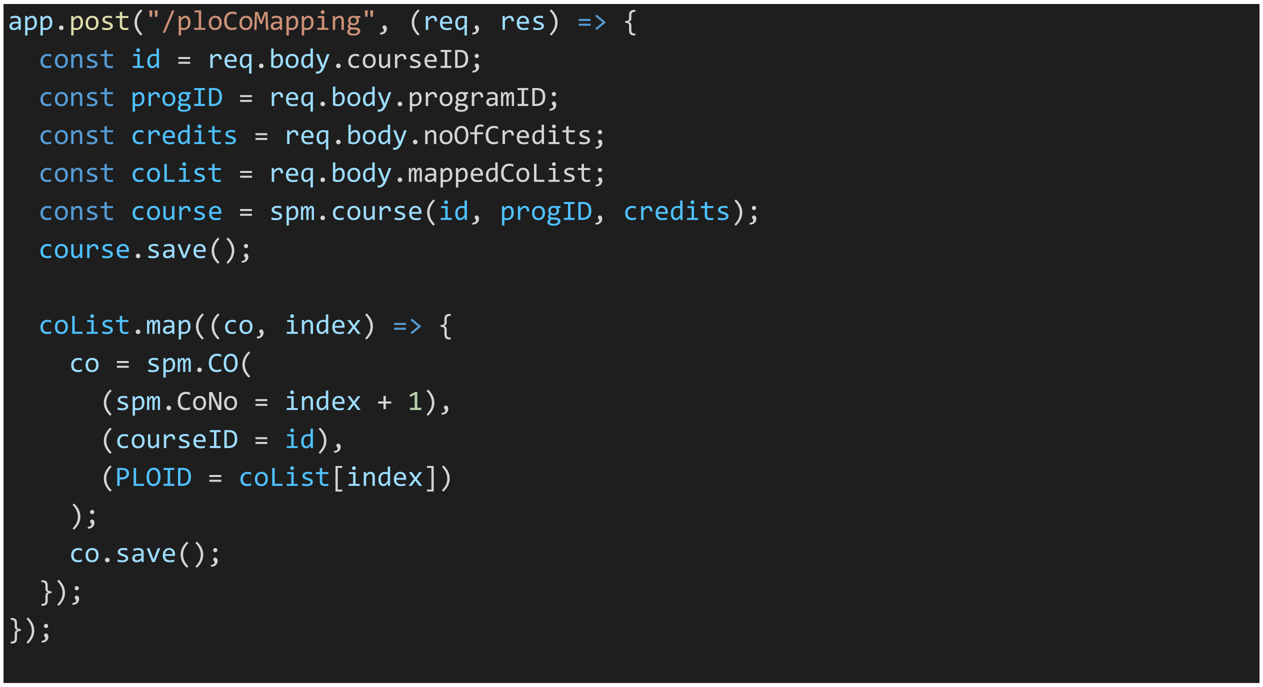
PLO-CO Mapping

CO1 maps To: PLO1

CO2 maps To: PLO2

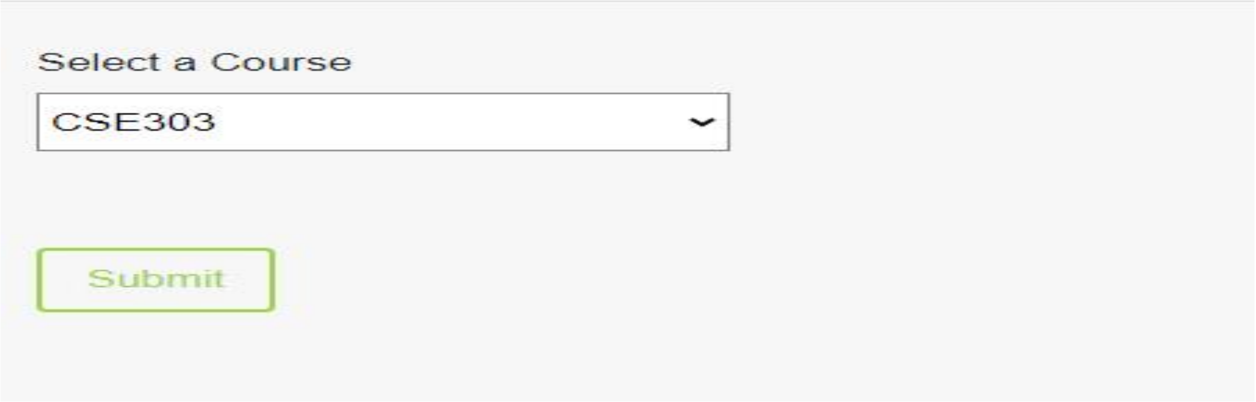
CO3 maps To: PLO3

CO4 maps To: PLO4



```
app.post("/ploCoMapping", (req, res) => {
  const id = req.body.courseID;
  const progID = req.body.programID;
  const credits = req.body.noOfCredits;
  const coList = req.body.mappedCoList;
  const course = spm.course(id, progID, credits);
  course.save();

  coList.map((co, index) => {
    co = spm.CO(
      (spm.CoNo = index + 1),
      (courseID = id),
      (PLOID = coList[index])
    );
    co.save();
  });
});
```

Couse-wise Student GPA

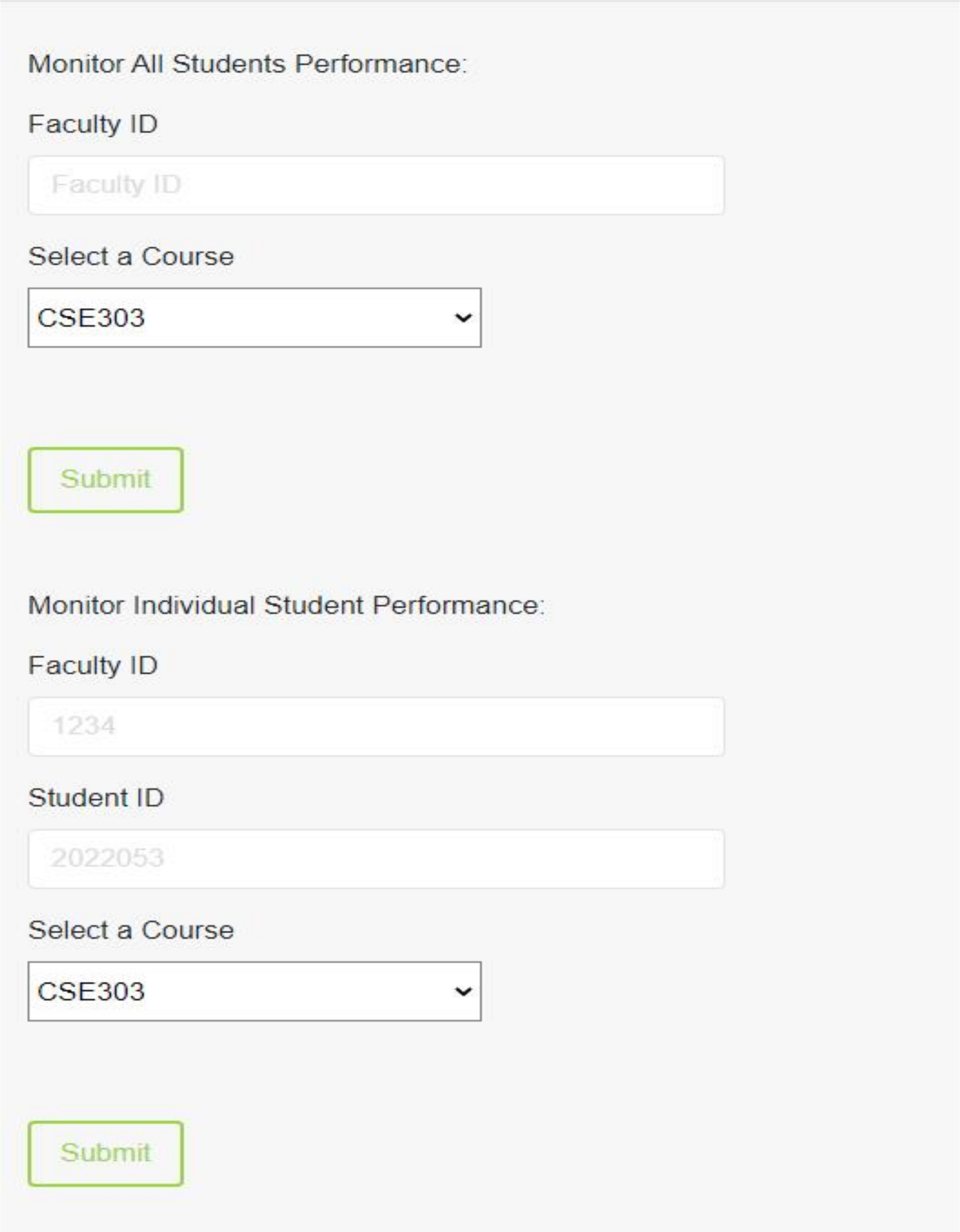


Select a Course: CSE303

```
app.post("/course/studentGrades", (req, res) => {
  var c = JSON.stringify(req.body.course);
  const grades = `SELECT R.studentID, R.courseName, R.grade
  FROM spm.course_grade R
  WHERE R.courseName = ${c}`;

  db.query(grades, (err, result) => {
    res.send(result);
  });
});
```

Faculty-wise Student GPA

```

app.post("/faculty/studentGrades", (req, res) => {
  const sqlInsert = `
  SELECT f.facultyId, CONCAT(f.first_name, " ", f.last_name) AS FacultyName,
  s.courseName, s.studentID, s.grade
  FROM spm.faculty f
  LEFT JOIN spm.course_grade s ON f.facultyID = s.facultyID
  WHERE f.facultyID = ${req.body.facultyID}`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});

app.post("/faculty/studentGrade", (req, res) => {
  const sqlInsert = `SELECT f.facultyId, CONCAT(f.first_name, " ", f.last_name) A
S FacultyName,
  s.courseName, s.studentID, s.grade
  FROM spm.faculty f
  LEFT JOIN spm.course_grade s ON f.facultyID = s.facultyID
  WHERE f.facultyID = ${req.body.facultyID} AND s.studentID = ${req.body.studentI
D};`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```



Monitor All Students Performance:

Faculty ID

Faculty ID

Select a Course

CSE303

Monitor Individual Student Performance:

Faculty ID

1234

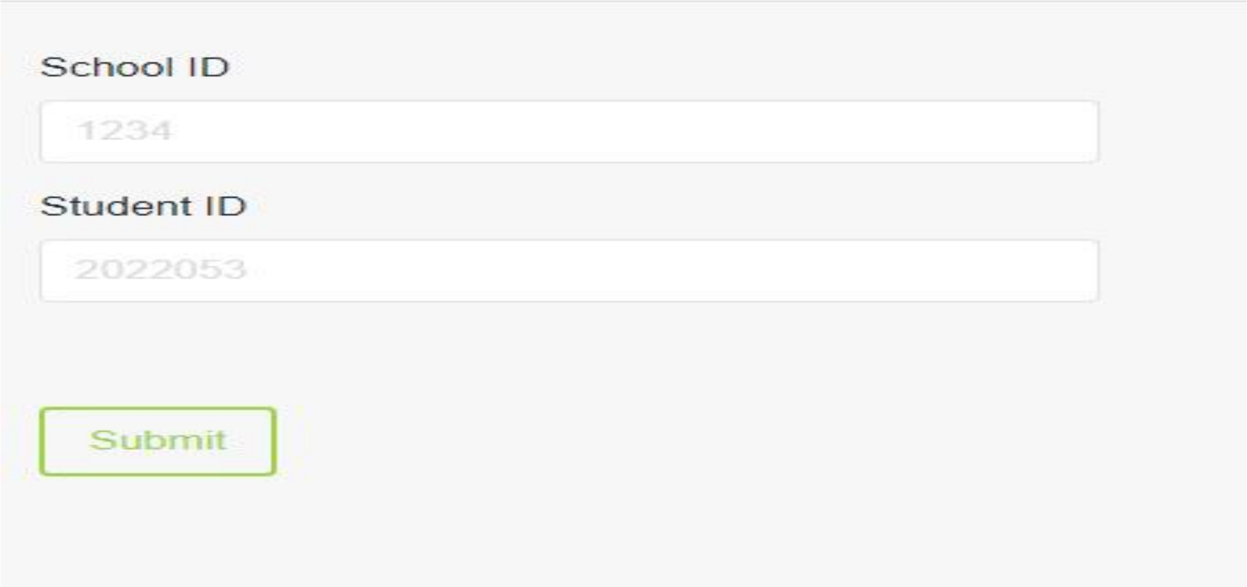
Student ID

2022053

Select a Course

CSE303

Dean-wise Student GPA

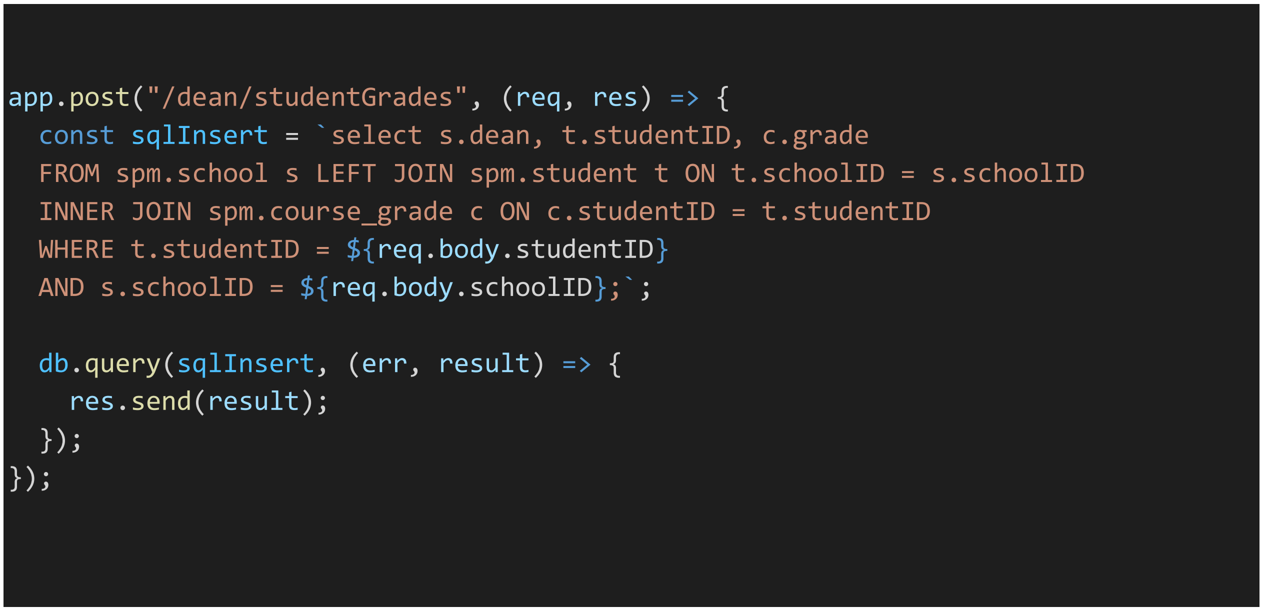


School ID

1234

Student ID

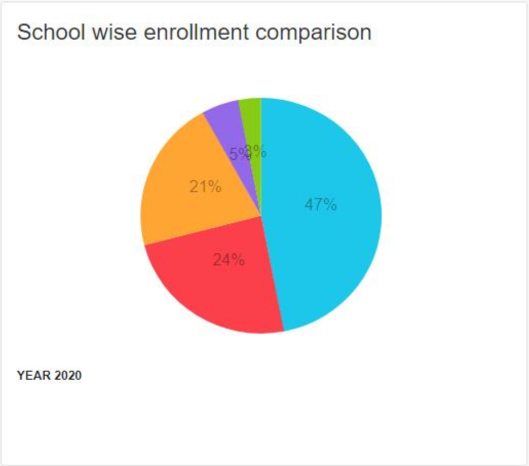
2022053

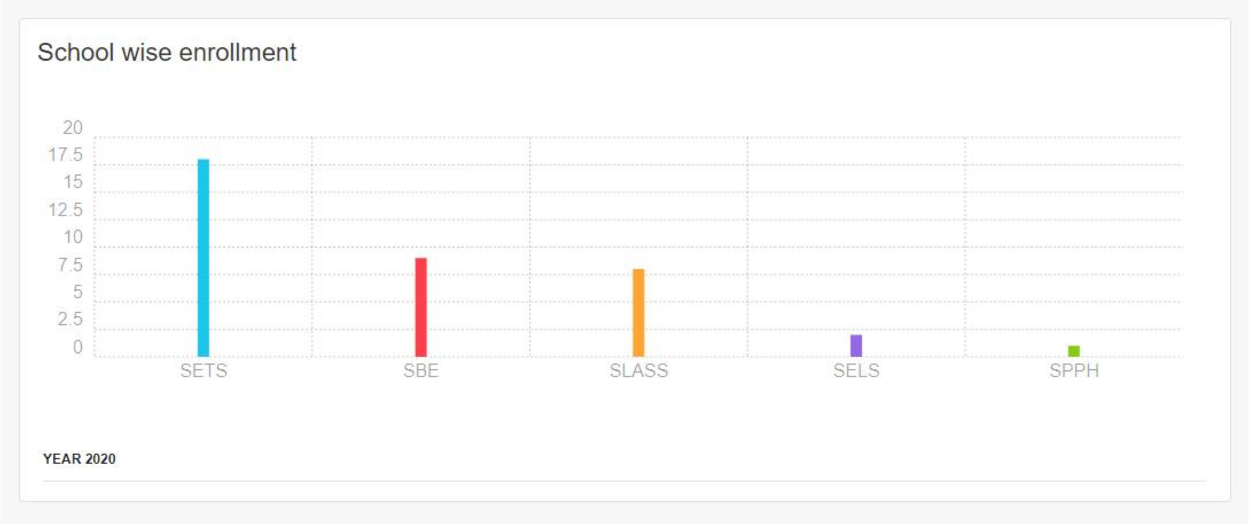


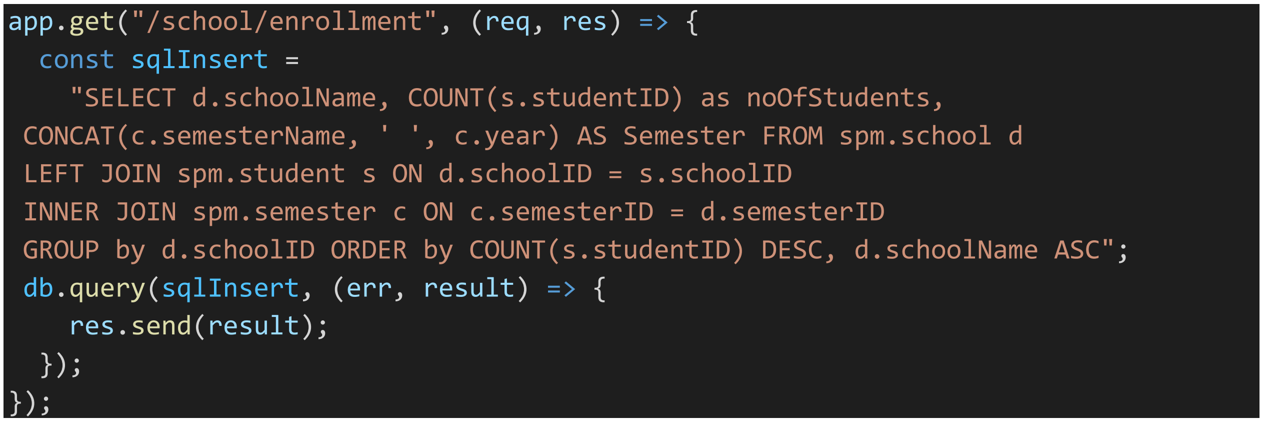
```
app.post("/dean/studentGrades", (req, res) => {
  const sqlInsert = `select s.dean, t.studentID, c.grade
  FROM spm.school s LEFT JOIN spm.student t ON t.schoolID = s.schoolID
  INNER JOIN spm.course_grade c ON c.studentID = t.studentID
  WHERE t.studentID = ${req.body.studentID}
  AND s.schoolID = ${req.body.schoolID};`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

OUTPUT FORMS

School-wise Enrollment

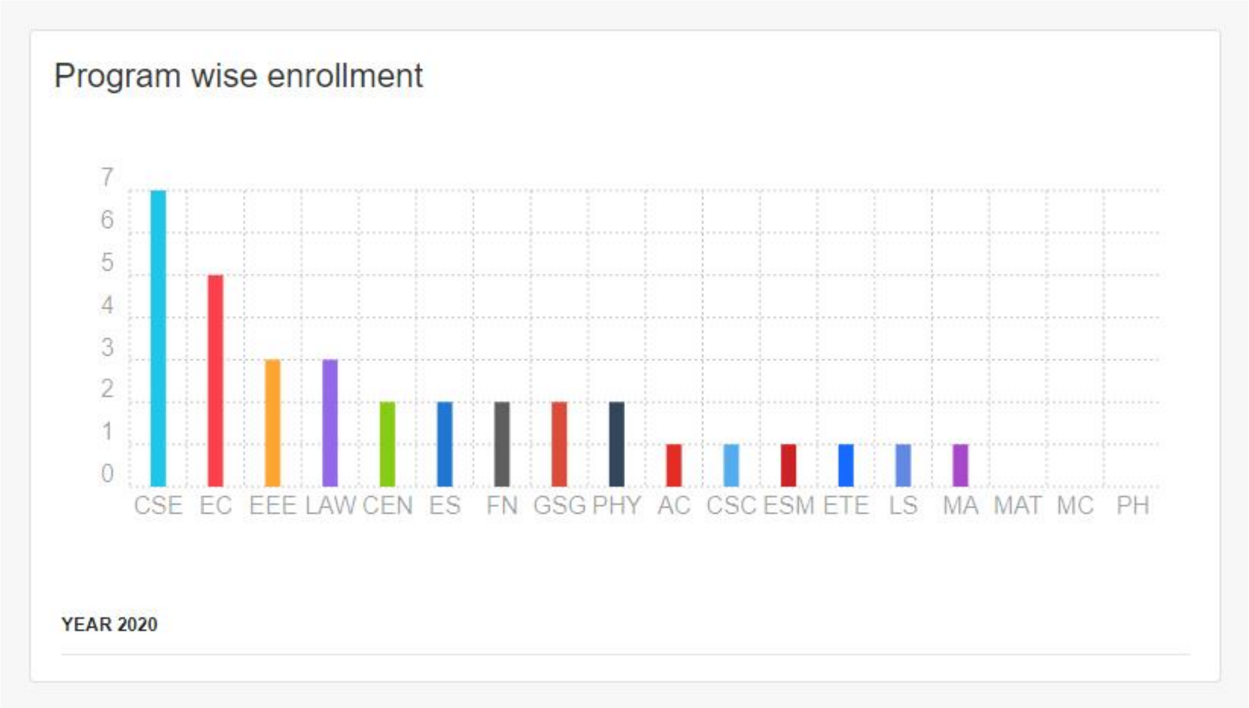


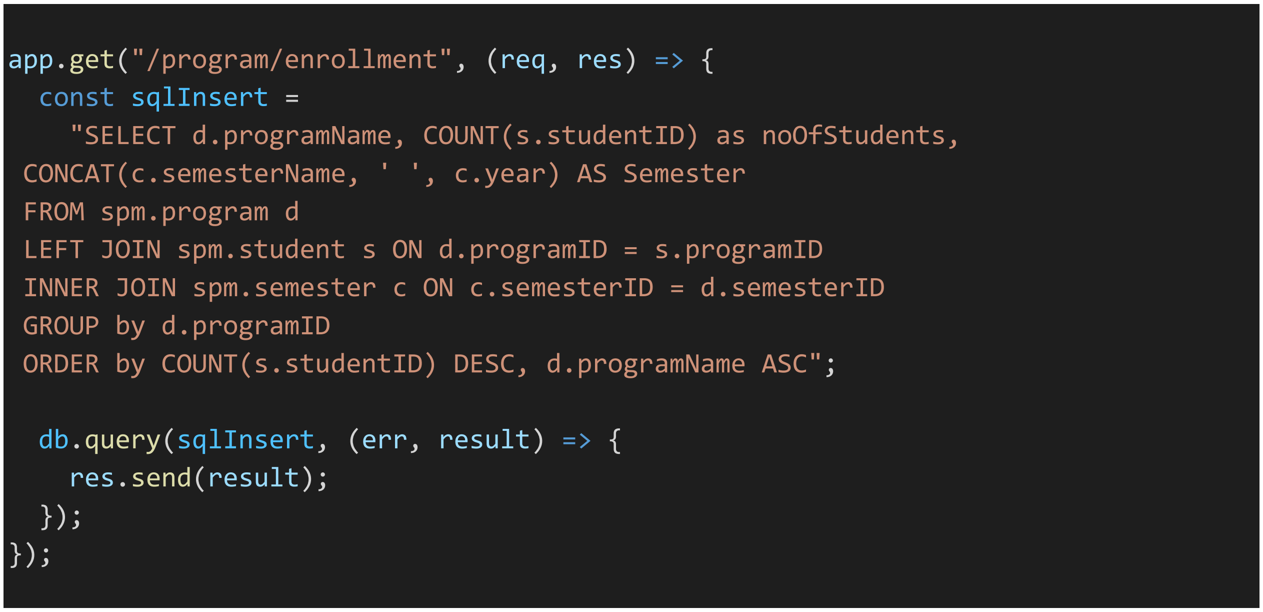




```
app.get("/school/enrollment", (req, res) => {
  const sqlInsert =
    "SELECT d.schoolName, COUNT(s.studentID) as noOfStudents,
 CONCAT(c.semesterName, ' ', c.year) AS Semester FROM spm.school d
 LEFT JOIN spm.student s ON d.schoolID = s.schoolID
 INNER JOIN spm.semester c ON c.semesterID = d.semesterID
 GROUP by d.schoolID ORDER by COUNT(s.studentID) DESC, d.schoolName ASC";
 db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

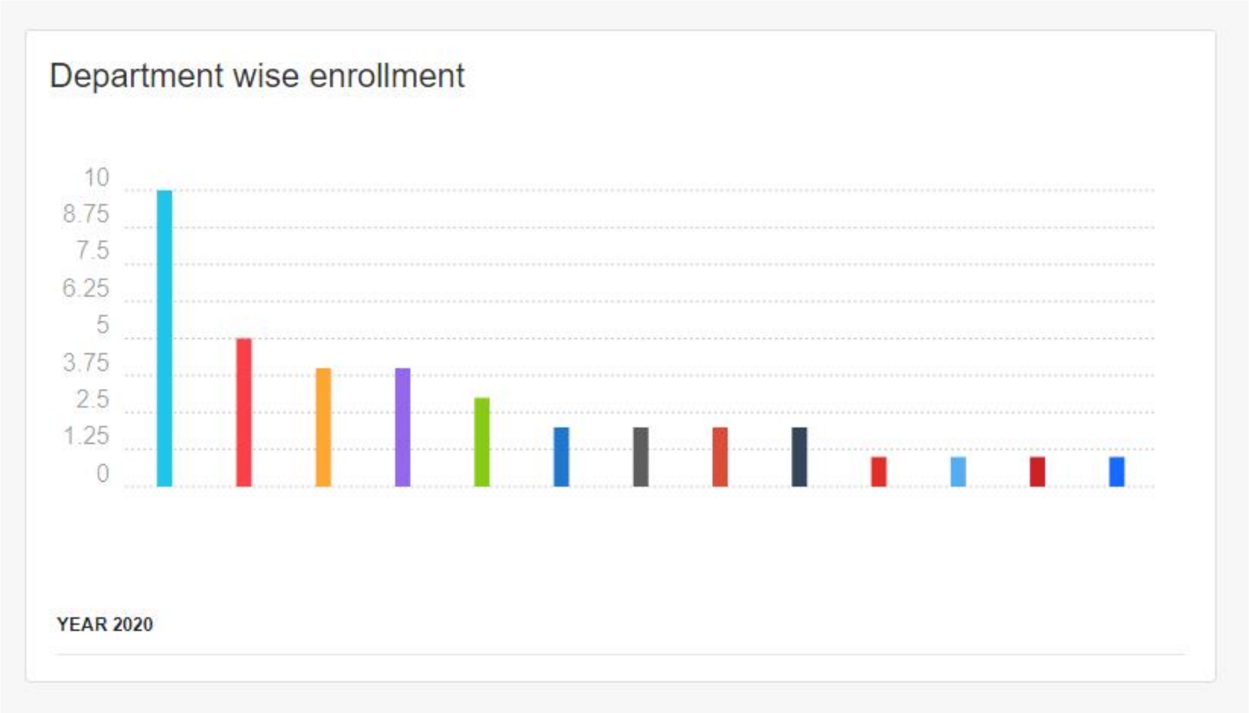
Program-wise Enrollment

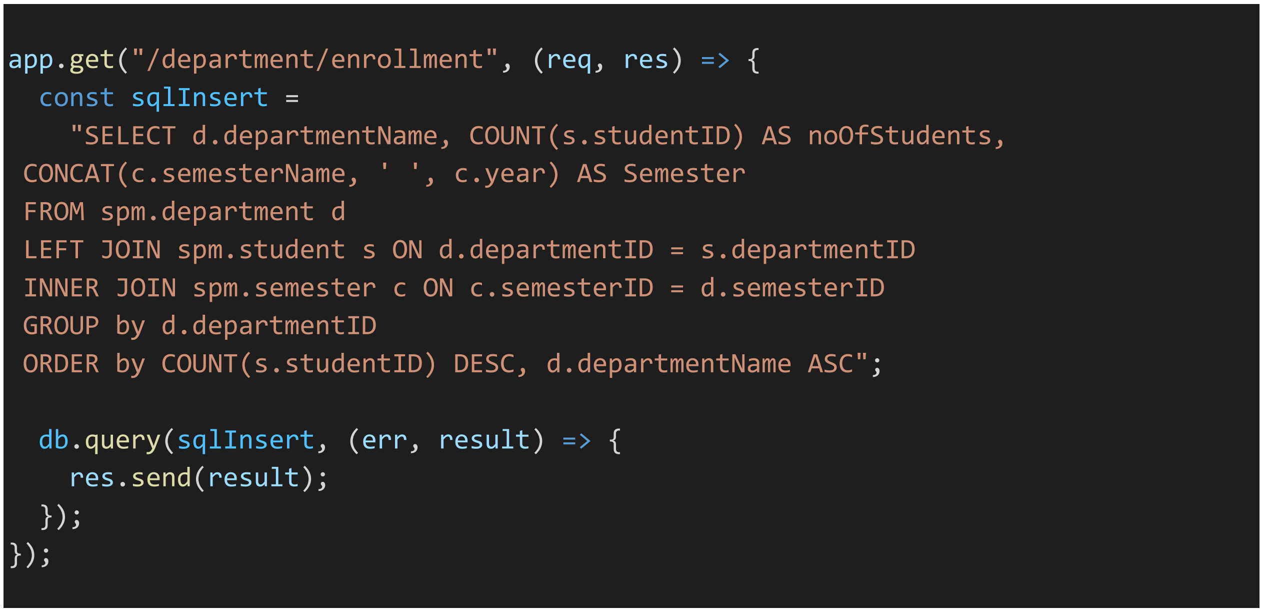




```
app.get("/program/enrollment", (req, res) => {
  const sqlInsert =
    "SELECT d.programName, COUNT(s.studentID) as noOfStudents,
 CONCAT(c.semesterName, ' ', c.year) AS Semester
 FROM spm.program d
 LEFT JOIN spm.student s ON d.programID = s.programID
 INNER JOIN spm.semester c ON c.semesterID = d.semesterID
 GROUP by d.programID
 ORDER by COUNT(s.studentID) DESC, d.programName ASC";

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

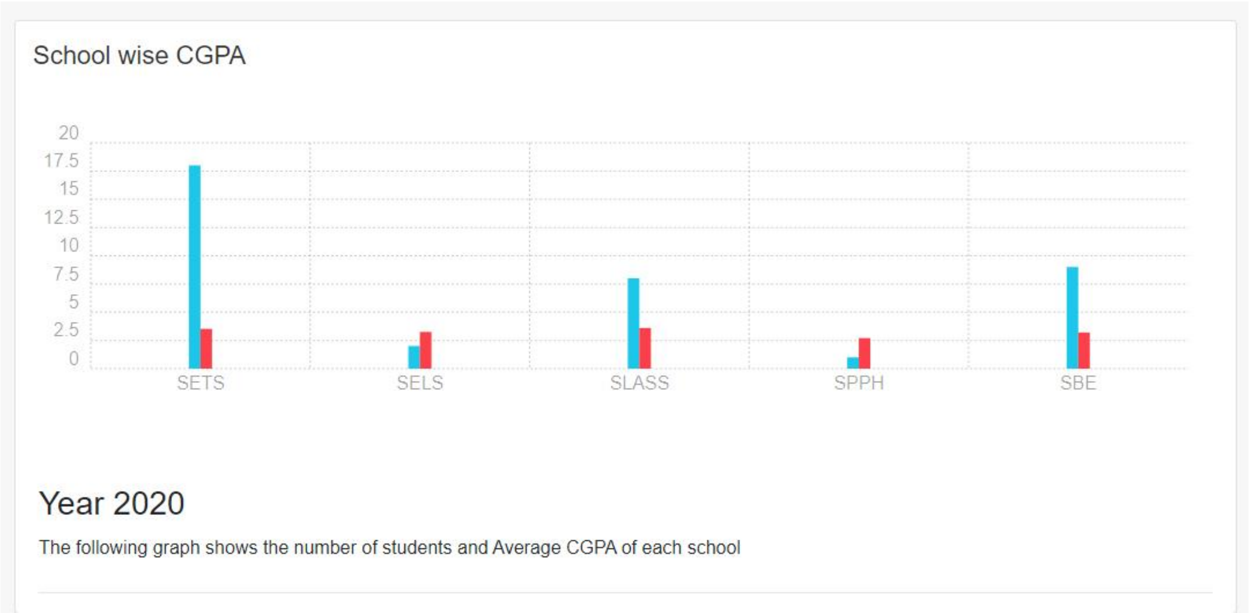
Department-wise Enrollment





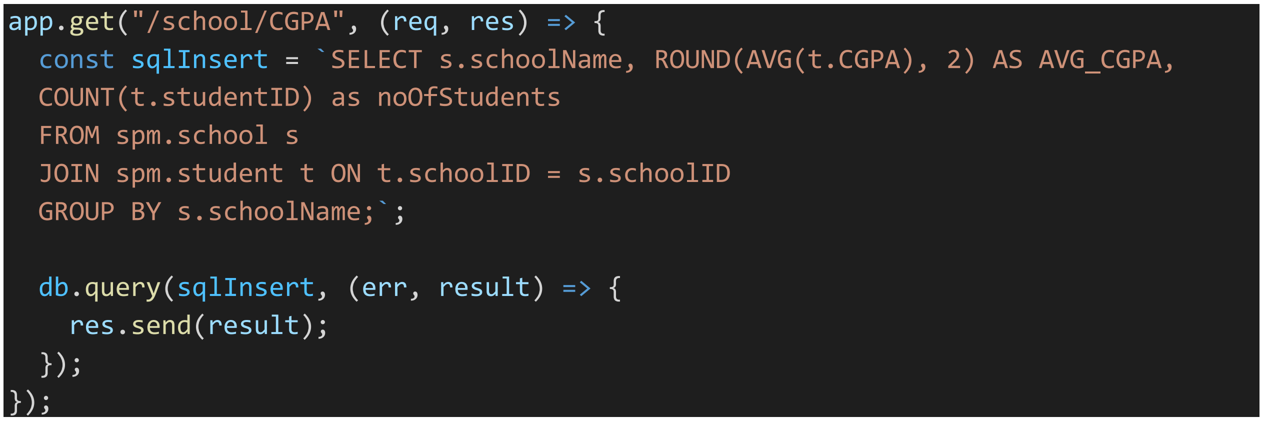
```
app.get("/department/enrollment", (req, res) => {
  const sqlInsert =
    "SELECT d.departmentName, COUNT(s.studentID) AS noOfStudents,
 CONCAT(c.semesterName, ' ', c.year) AS Semester
 FROM spm.department d
 LEFT JOIN spm.student s ON d.departmentID = s.departmentID
 INNER JOIN spm.semester c ON c.semesterID = d.semesterID
 GROUP by d.departmentID
 ORDER by COUNT(s.studentID) DESC, d.departmentName ASC";

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

School-wise CGPA



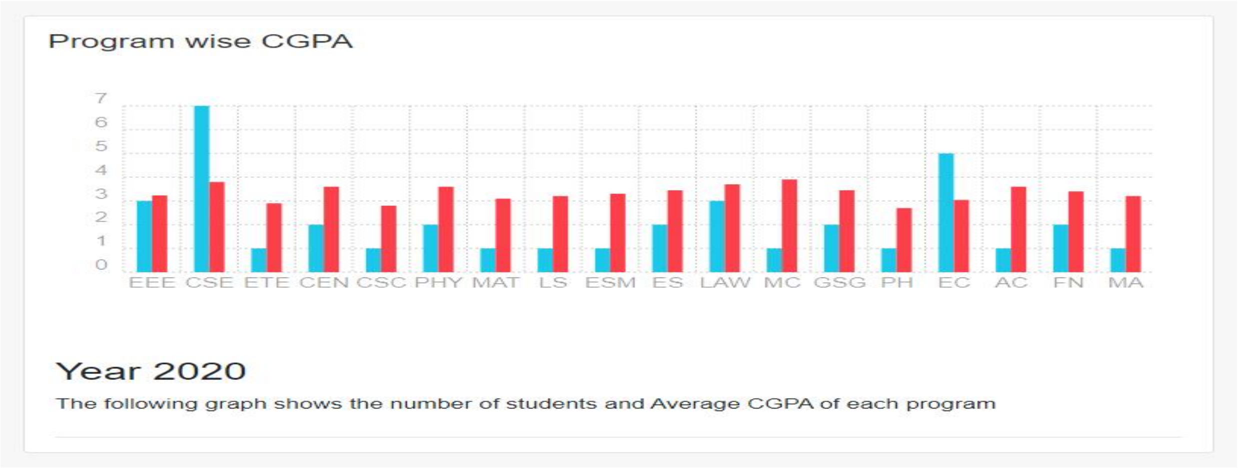
Year 2020

The following graph shows the number of students and Average CGPA of each school



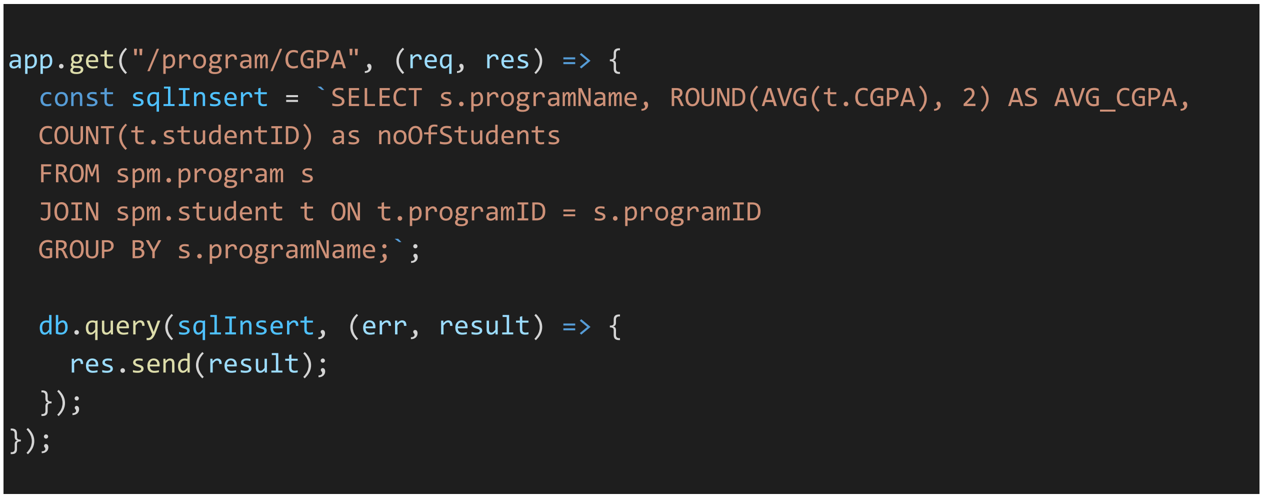
```
app.get("/school/CGPA", (req, res) => {
  const sqlInsert = `SELECT s.schoolName, ROUND(AVG(t.CGPA), 2) AS AVG_CGPA,
  COUNT(t.studentID) as noOfStudents
  FROM spm.school s
  JOIN spm.student t ON t.schoolID = s.schoolID
  GROUP BY s.schoolName;`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

Program-wise CGPA



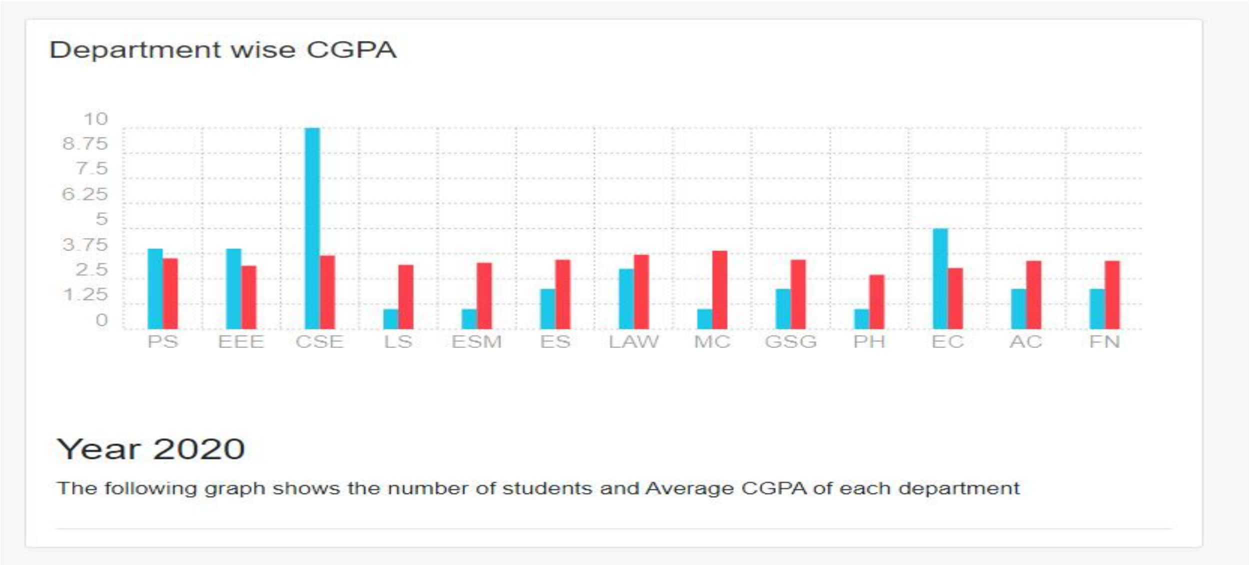
Year 2020

The following graph shows the number of students and Average CGPA of each program



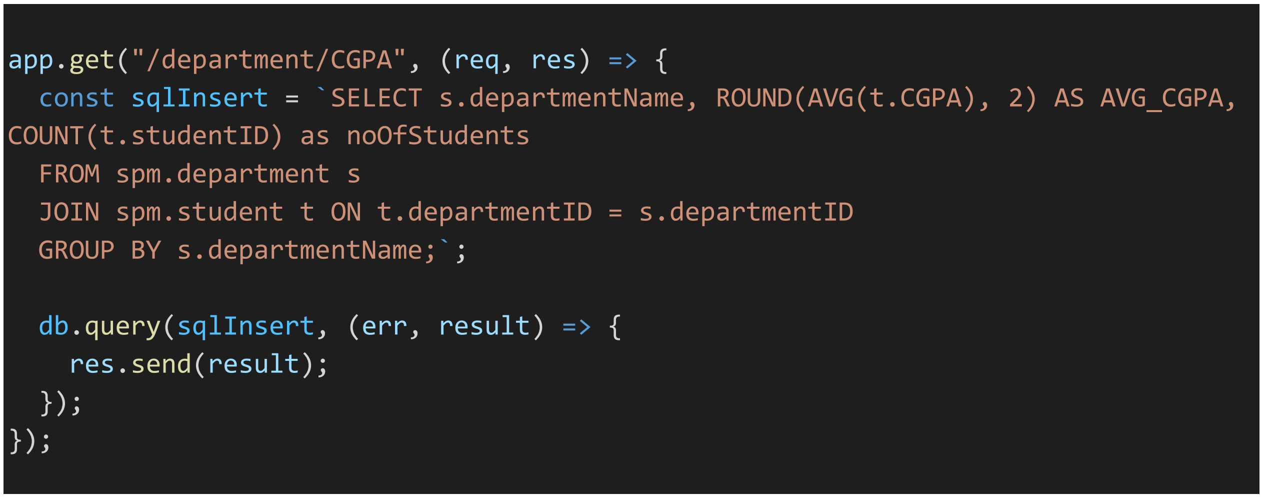
```
app.get("/program/CGPA", (req, res) => {
  const sqlInsert = `SELECT s.programName, ROUND(AVG(t.CGPA), 2) AS AVG_CGPA,
  COUNT(t.studentID) as noOfStudents
  FROM spm.program s
  JOIN spm.student t ON t.programID = s.programID
  GROUP BY s.programName;`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

Department-wise CGPA



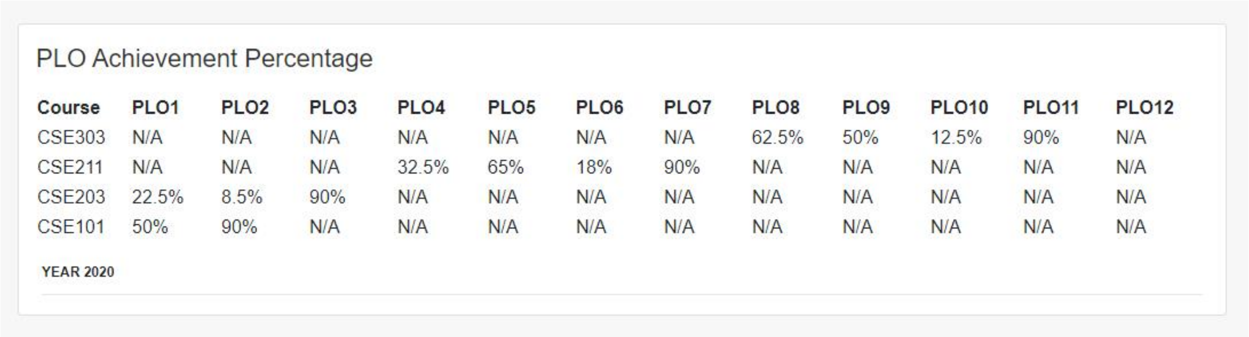
Year 2020

The following graph shows the number of students and Average CGPA of each department



```
app.get("/department/CGPA", (req, res) => {
  const sqlInsert = `SELECT s.departmentName, ROUND(AVG(t.CGPA), 2) AS AVG_CGPA,
COUNT(t.studentID) as noOfStudents
  FROM spm.department s
  JOIN spm.student t ON t.departmentID = s.departmentID
  GROUP BY s.departmentName;`;

  db.query(sqlInsert, (err, result) => {
    res.send(result);
  });
});
```

PLO Achievement Percentage Table



PLO Achievement Percentage

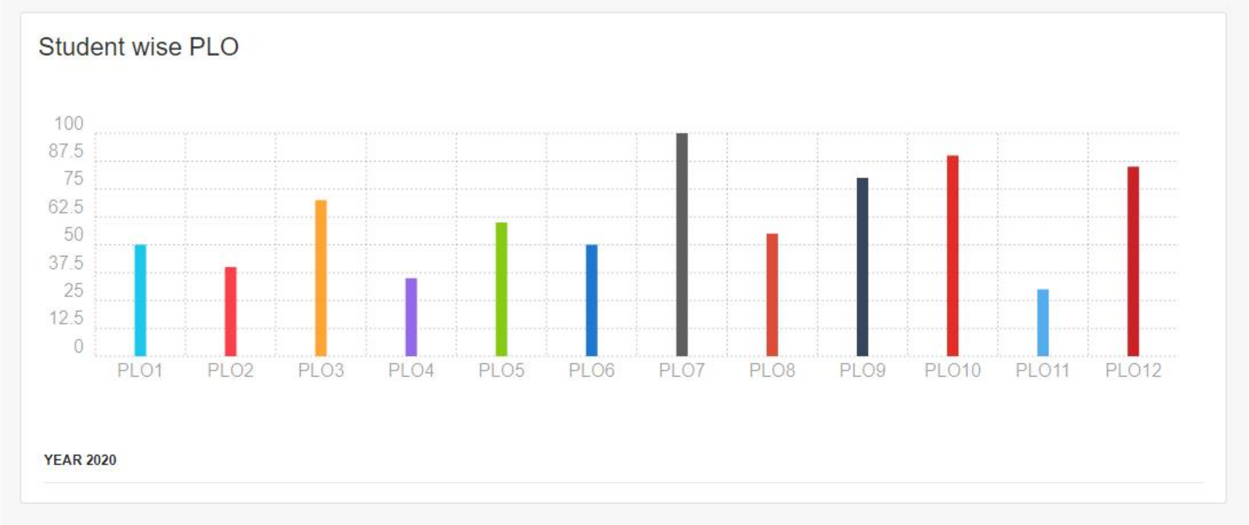
Course	PLO1	PLO2	PLO3	PLO4	PLO5	PLO6	PLO7	PLO8	PLO9	PLO10	PLO11	PLO12
CSE303	N/A	N/A	N/A	N/A	N/A	N/A	N/A	62.5%	50%	12.5%	90%	N/A
CSE211	N/A	N/A	N/A	32.5%	65%	18%	90%	N/A	N/A	N/A	N/A	N/A
CSE203	22.5%	8.5%	90%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
CSE101	50%	90%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

YEAR 2020

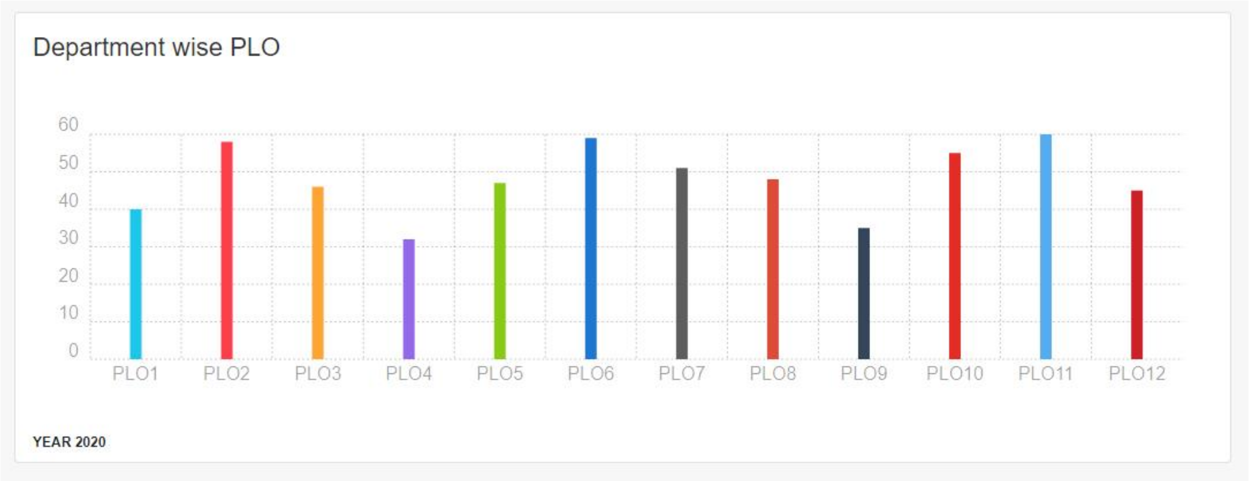
```
SELECT DISTINCT co.courseID, co.CONo, p.PLONo, (PLO / CoTotal * 100) AS
PLOpercentage
FROM spm.PLO p, spm.CO co, (
SELECT DISTINCT c.courseID, c.CONo, c.PLOID, SUM(DISTINCT
e.marksObtained) AS PLO, SUM(DISTINCT a.marksObtained) AS CoTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.studentID = 2022053
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND c.PLOID = p.PLONO
AND a.COID = c.COID
GROUP BY en.sectionID, c.PLOID
ORDER BY c.PLOID
) perPLO
WHERE co.CONo = perPLO.CONo
AND p.PLONo = perPLO.PLOID
AND co.courseID = perPLO.courseID;
```

Student-wise PLO

```
SELECT AVG(TotalPlo.PLOpercentage) AS ActualPlo
FROM (
SELECT (PLO / COTotal * 100) AS PLOpercentage
FROM (
SELECT SUM(DISTINCT e.marksObtained) AS PLO, SUM(DISTINCT
a.marksObtained) AS CoTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.co c,
spm.plo p
WHERE en.studentID = {}
AND e.assessmentID = a.assessmentID
AND en.enrollmentID = e.enrollmentID
AND a.COID = c.COID
AND c.PLOID = {}
GROUP BY en.sectionID
) perPLO
) TotalPlo;
```

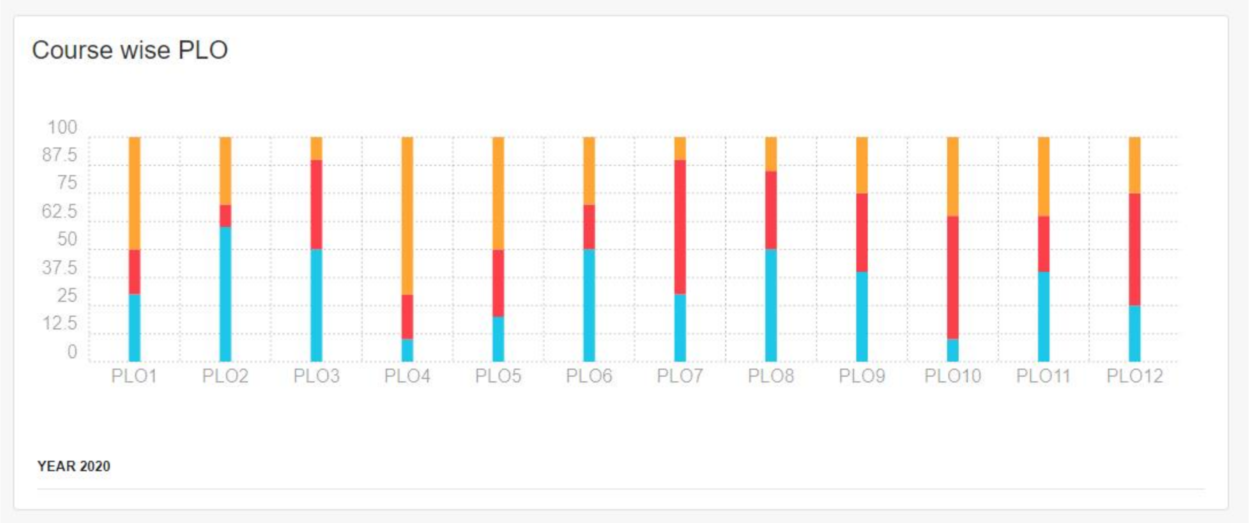


Department-wise PLO



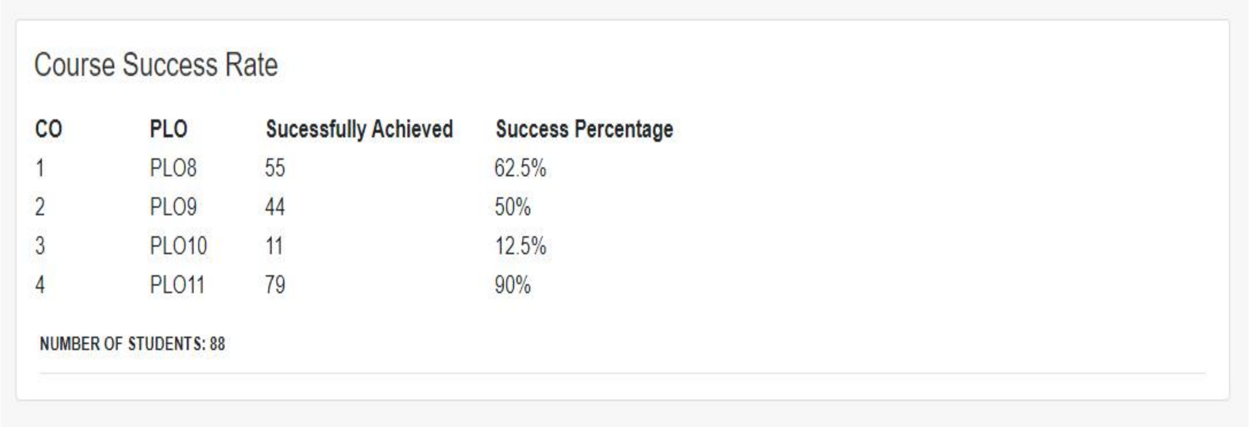
```
SELECT AVG(PLOTotal.PLOpercentage) AS ActualPlo
FROM (
SELECT (PLO / COTotal * 100) AS PLOpercentage
FROM (
SELECT SUM(e.marksObtained) AS PLO, SUM(a.marksObtained) AS
COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.co c,
spm.plo p,
spm.student st
WHERE st.departmentID = 100
AND st.studentID = en.studentID
AND e.assessmentID = a.assessmentID
AND en.enrollmentID = e.enrollmentID
AND a.COID = c.COID
AND c.PLOID = 1
GROUP BY en.sectionID
) perPLO
) PLOTotal;
```

Course-wise PLO:



```
SELECT DISTINCT co.courseID, co.CONo, p.PLONo, (PLO / TotalCO * 100) AS
PLOpercentage
FROM spm.PLO p, spm.CO co, (
SELECT DISTINCT c.courseID, c.CONo, c.PLOID, SUM(DISTINCT
e.marksObtained) AS PLO,
SUM(DISTINCT a.marksObtained) AS TotalCO
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.studentID = 2022053
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = p.PLONo
GROUP BY en.sectionID, c.PLOID
ORDER BY c.PLOID
) perPLO
WHERE CO.CONo = perPLO.CONo
AND CO.courseID = perPLO.courseID
AND p.PLONo = perPLO.PLOID;
```

Course Success Rate

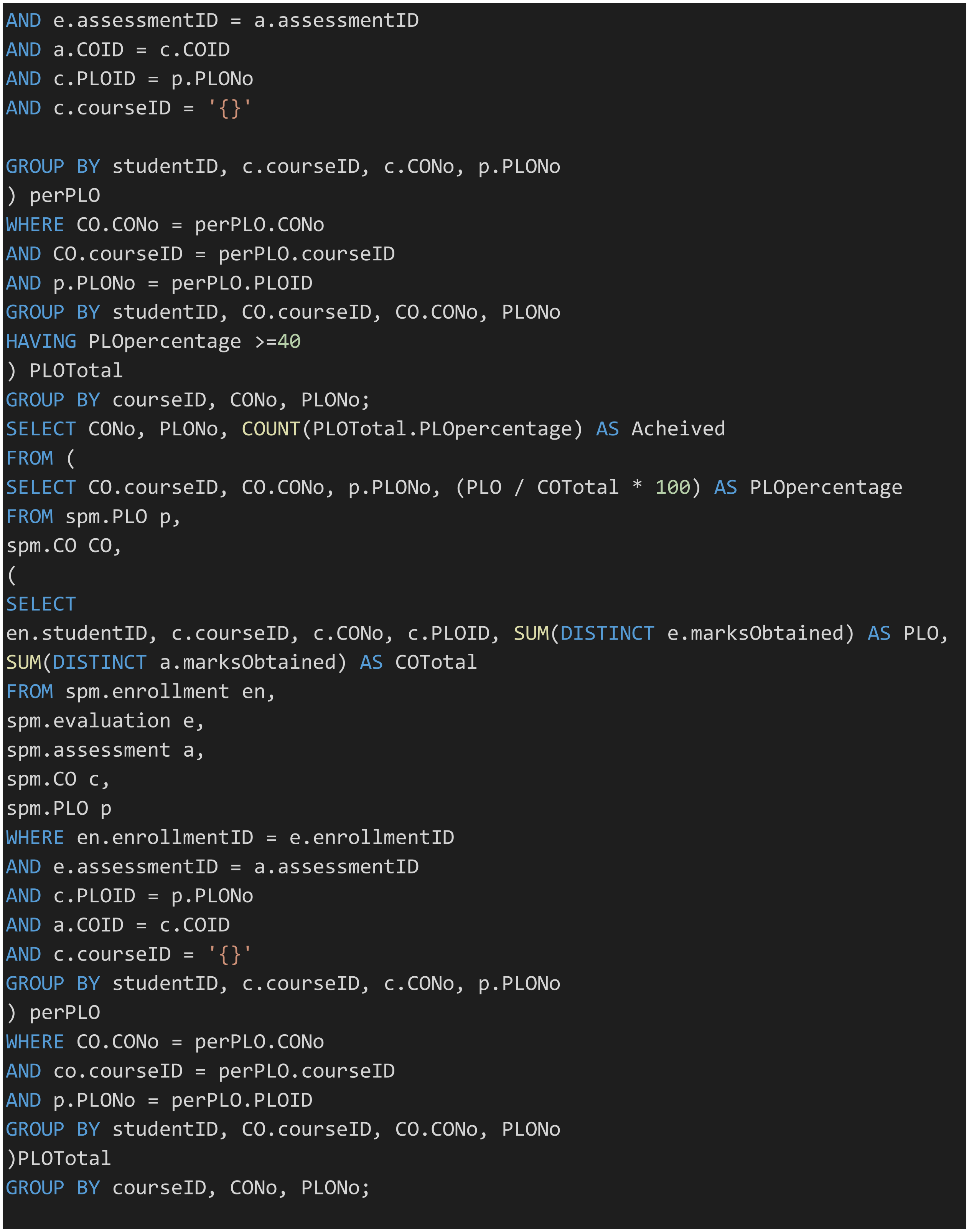


Course Success Rate

CO	PLO	Sucessfully Achieved	Success Percentage
1	PLO8	55	62.5%
2	PLO9	44	50%
3	PLO10	11	12.5%
4	PLO11	79	90%

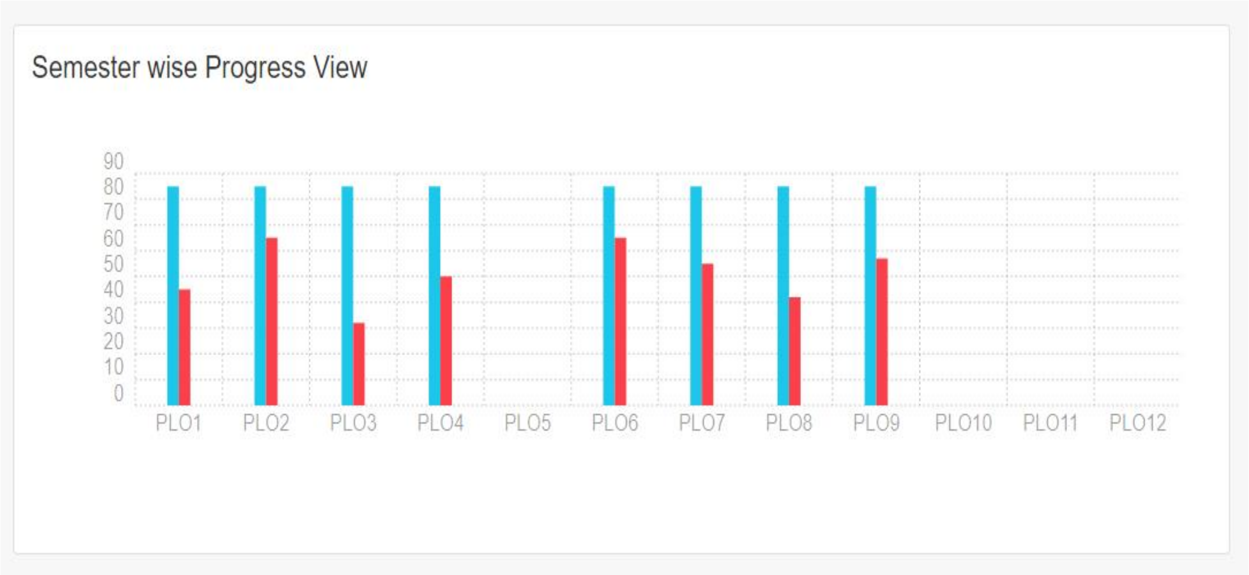
NUMBER OF STUDENTS: 88

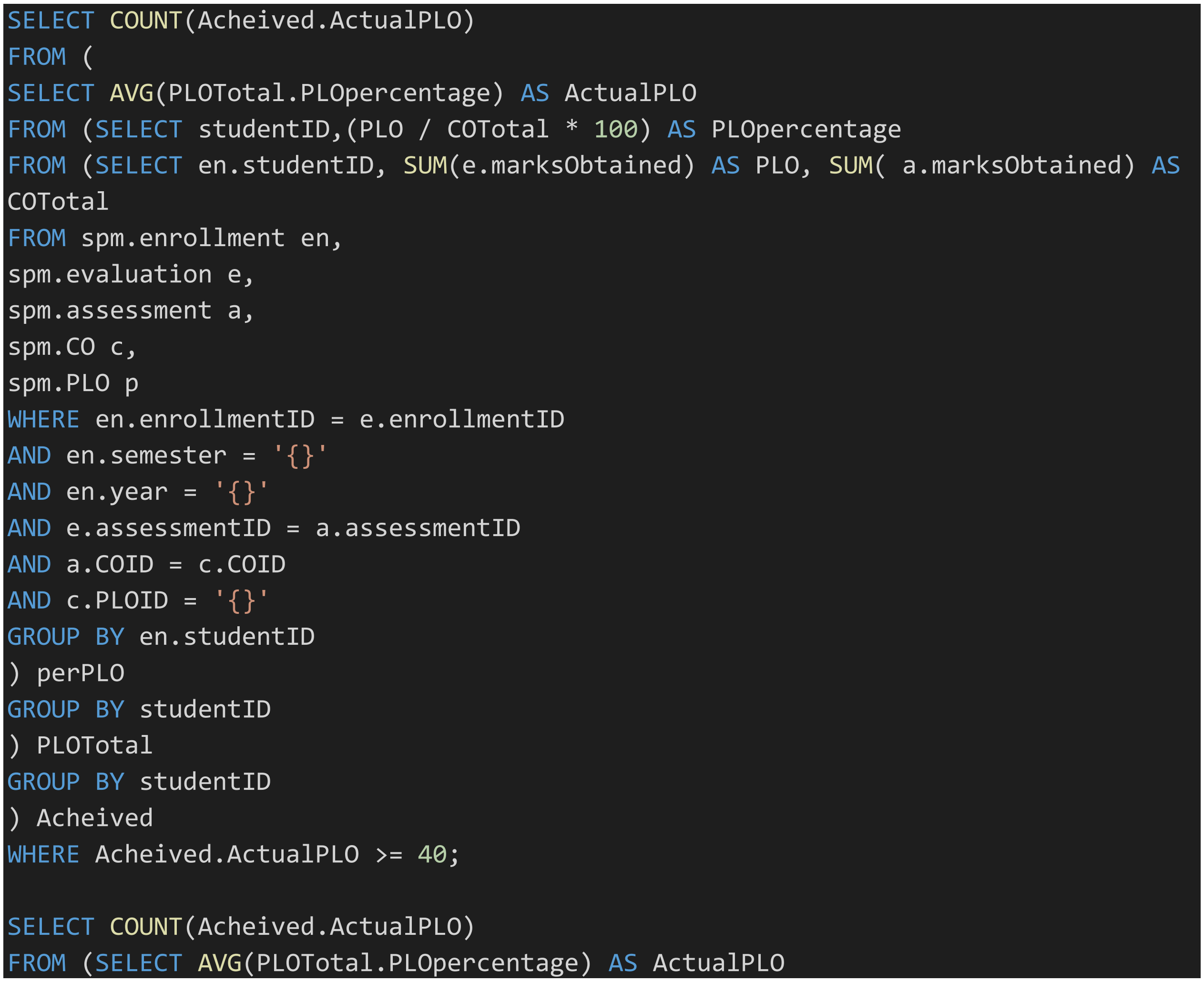
```
SELECT CONo, PLONo, COUNT(PLOTotal.PLOpercentage) AS Acheived
FROM (
SELECT CO.courseID, CO.CONo, p.PLONo,(PLO / COTotal * 100)
AS PLOpercentage
FROM spm.PLO p,
spm.CO CO,
(
SELECT
en.studentID, c.courseID, c.CONo, c.PLOID, SUM(DISTINCT e.marksObtained) AS
PLO, SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.enrollmentID = e.enrollmentID
```



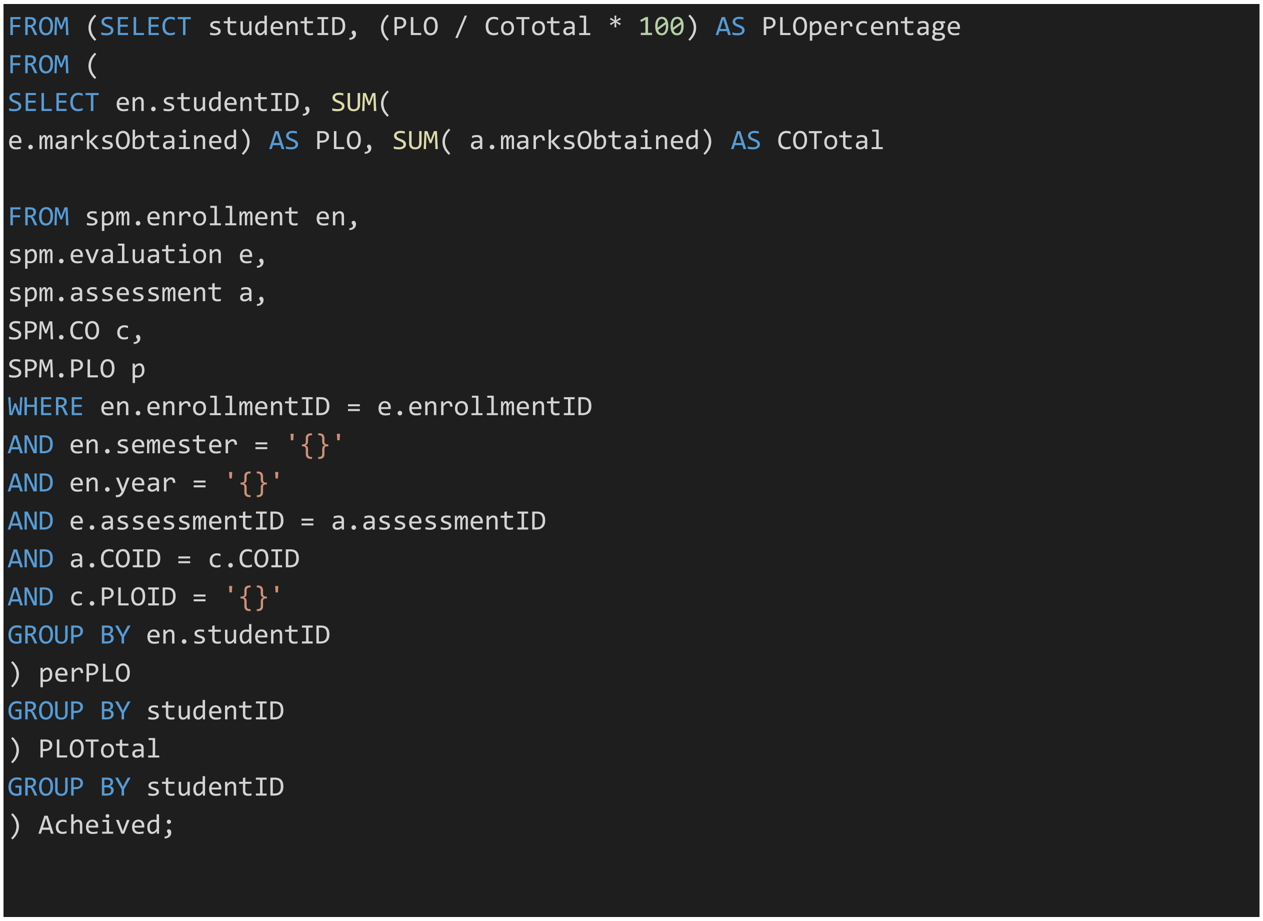
```
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = p.PLONo
AND c.courseID = '{}'

GROUP BY studentID, c.courseID, c.CONo, p.PLONo
) perPLO
WHERE CO.CONo = perPLO.CONo
AND CO.courseID = perPLO.courseID
AND p.PLONo = perPLO.PLOID
GROUP BY studentID, CO.courseID, CO.CONo, PLONo
HAVING PLOpercentage >=40
) PLOTotal
GROUP BY courseID, CONo, PLONo;
SELECT CONo, PLONo, COUNT(PLOTotal.PLOpercentage) AS Acheived
FROM (
SELECT CO.courseID, CO.CONo, p.PLONo, (PLO / COTotal * 100) AS PLOpercentage
FROM spm.PLO p,
spm.CO CO,
(
SELECT
en.studentID, c.courseID, c.CONo, c.PLOID, SUM(DISTINCT e.marksObtained) AS PLO,
SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND c.PLOID = p.PLONo
AND a.COID = c.COID
AND c.courseID = '{}'
GROUP BY studentID, c.courseID, c.CONo, p.PLONo
) perPLO
WHERE CO.CONo = perPLO.CONo
AND co.courseID = perPLO.courseID
AND p.PLONo = perPLO.PLOID
GROUP BY studentID, CO.courseID, CO.CONo, PLONo
)PLOTotal
GROUP BY courseID, CONo, PLONo;
```

Semester-wise Progress:



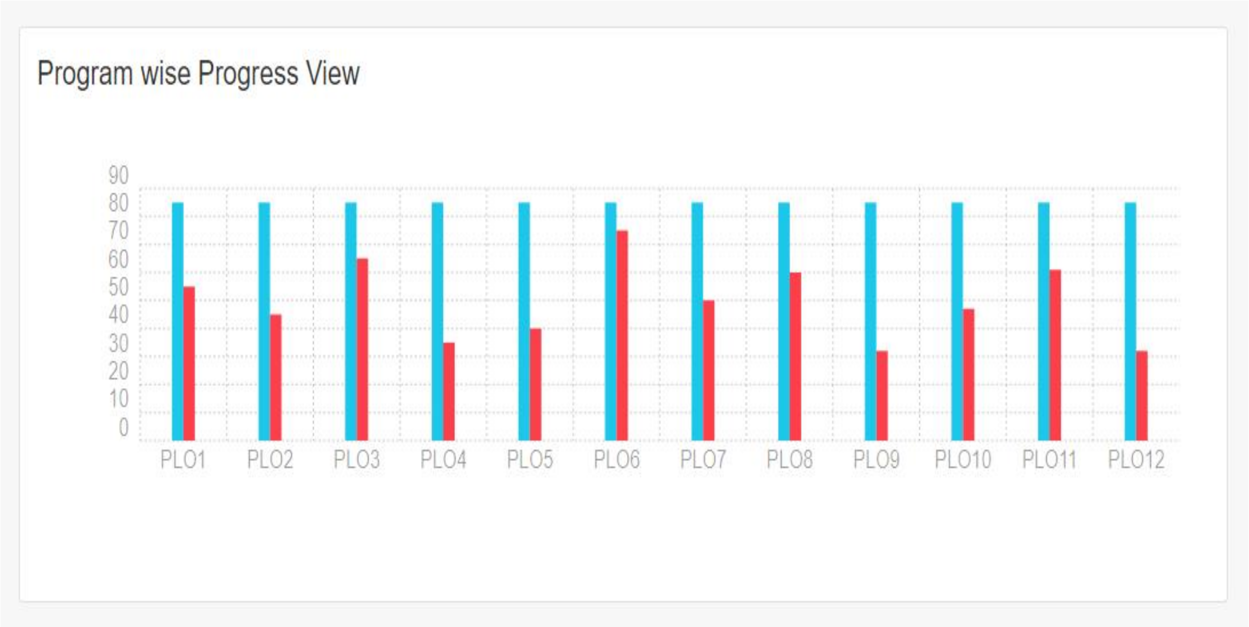


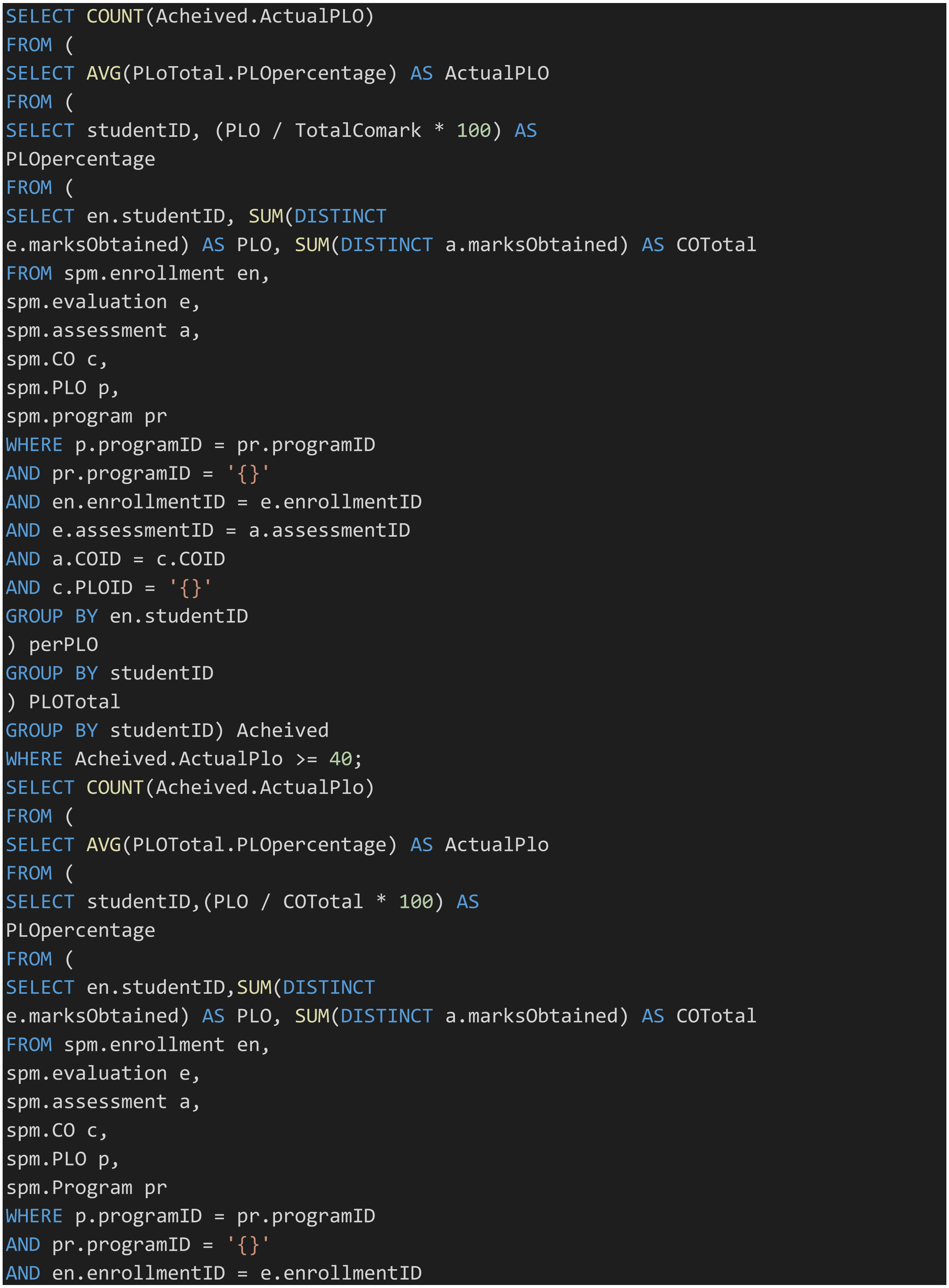
```sql
SELECT COUNT(Acheived.ActualPLO)
FROM (
SELECT AVG(PLOTotal.PLOpercentage) AS ActualPLO
FROM (SELECT studentID,(PLO / COTotal * 100) AS PLOpercentage
FROM (SELECT en.studentID, SUM(e.marksObtained) AS PLO, SUM( a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.enrollmentID = e.enrollmentID
AND en.semester = '{}'
AND en.year = '{}'
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = '{}'
GROUP BY en.studentID
) perPLO
GROUP BY studentID
) PLOTotal
GROUP BY studentID
) Acheived
WHERE Acheived.ActualPLO >= 40;

SELECT COUNT(Acheived.ActualPLO)
FROM (SELECT AVG(PLOTotal.PLOpercentage) AS ActualPLO
```



```
FROM (SELECT studentID, (PLO / CoTotal * 100) AS PLOpercentage
FROM (
SELECT en.studentID, SUM(
e.marksObtained) AS PLO, SUM( a.marksObtained) AS COTotal

FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
SPM.CO c,
SPM.PLO p
WHERE en.enrollmentID = e.enrollmentID
AND en.semester = '{}'
AND en.year = '{}'
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = '{}'
GROUP BY en.studentID
) perPLO
GROUP BY studentID
) PLOTotal
GROUP BY studentID
) Acheived;
```

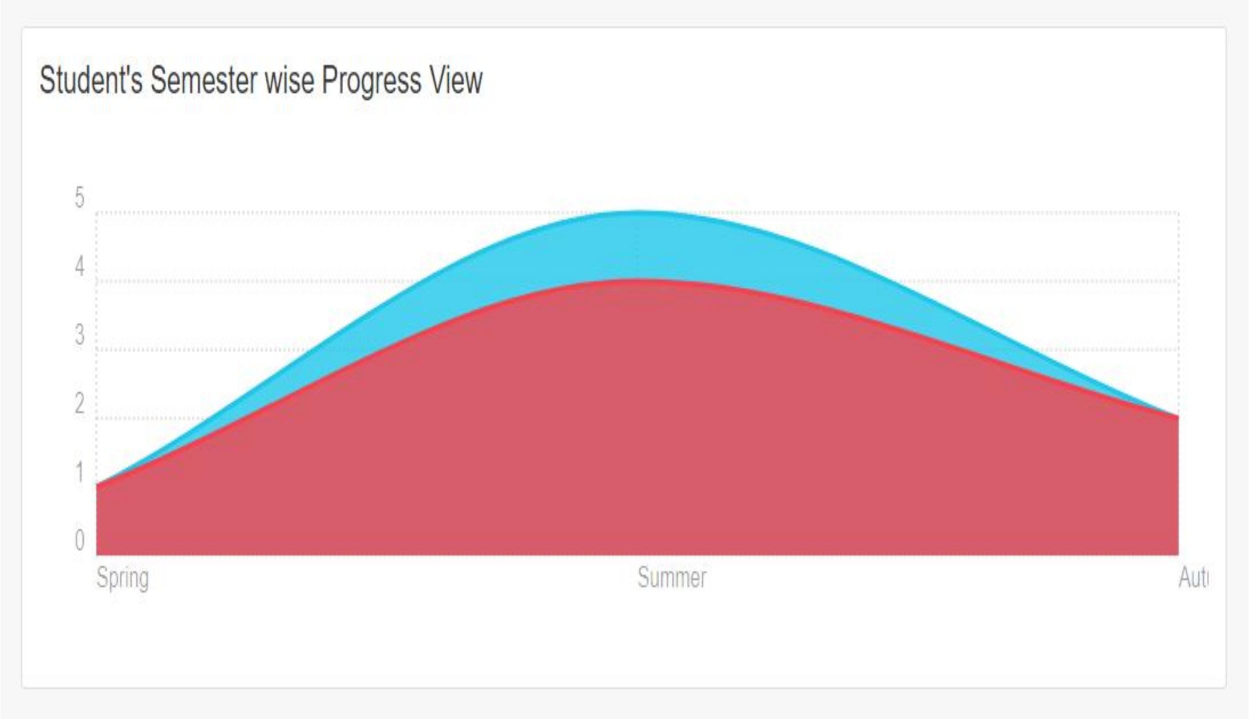
Program-wise Progress





```
SELECT COUNT(Acheived.ActualPLO)
FROM (
SELECT AVG(PLoTotal.PLOpercentage) AS ActualPLO
FROM (
SELECT studentID, (PLO / TotalComark * 100) AS
PLOpercentage
FROM (
SELECT en.studentID, SUM(DISTINCT
e.marksObtained) AS PLO, SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p,
spm.program pr
WHERE p.programID = pr.programID
AND pr.programID = '{}'
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = '{}'
GROUP BY en.studentID
) perPLO
GROUP BY studentID
) PLOTotal
GROUP BY studentID) Acheived
WHERE Acheived.ActualPlo >= 40;
SELECT COUNT(Acheived.ActualPlo)
FROM (
SELECT AVG(PLOTotal.PLOpercentage) AS ActualPlo
FROM (
SELECT studentID,(PLO / COTotal * 100) AS
PLOpercentage
FROM (
SELECT en.studentID,SUM(DISTINCT
e.marksObtained) AS PLO, SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p,
spm.Program pr
WHERE p.programID = pr.programID
AND pr.programID = '{}'
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = '{}'
GROUP BY en.studentID
) perPLO
GROUP BY studentID
) PLOTotal
GROUP BY studentID
) Acheived;
```

Student’ Semester-wise Progress



```
SELECT COUNT(Acheived.ActualPLO) AS isPLOAchieved
FROM(
SELECT AVG(PLOTotal.PLOpercentage) AS ActualPLO
FROM (
SELECT (PLO / COTotal * 100) AS PLOpercentage
FROM (
SELECT SUM(DISTINCT e.marksObtained) AS PLO,
SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
```

```sql
spm.PLO p

WHERE en.studentID = {}
AND en.semester = {}
AND en.year = {}
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = 1
GROUP BY en.semester
) perPLO
) PLOTotal
) Acheived
WHERE Acheived.ActualPlo >=40;

SELECT COUNT(Acheived.ActualPlo) AS isPLOAchieved
FROM(
SELECT AVG(PLOTotal.PLOpercentage) AS ActualPlo
FROM (
SELECT (PLO / COTotal * 100) AS PLOpercentage
FROM (
SELECT SUM(DISTINCT e.marksObtained) AS PLO,
SUM(DISTINCT a.marksObtained) AS COTotal
FROM spm.enrollment en,
spm.evaluation e,
spm.assessment a,
spm.CO c,
spm.PLO p
WHERE en.studentID = {}
AND en.semester = {}
AND en.year = {}
AND en.enrollmentID = e.enrollmentID
AND e.assessmentID = a.assessmentID
AND a.COID = c.COID
AND c.PLOID = 1
GROUP BY en.semester
) perPLO
) PLOTotal
) Acheived;
```