SNU-COMPASS

College **O**ptions and **M**inor **P**references **A**ssessment **S**upport **S**ystem

CSD317 (Introduction to Database Systems) Semester Project

Team members:

Name	Roll-Number
Rahul Jayaram	2110110410
Punyam Singh	2110110842
Aayush Arora	2110110017
Mitaali Singhal	2110110883

Objective

Our project SNU-COMPASS aims at developing a user-friendly web application to address a common challenge faced by SNU students, especially freshmen and sophomores. Many students lack clear guidance on how to pursue a Minor or Specialization alongside their major. Our app aims to bridge this knowledge gap by providing students with comprehensive information on required University Wide Electives (UWEs) for a Minor and Major Electives for a Specialization. Our goal is to simplify academic planning and empower all future SNU students to make informed decisions about their academic paths.

Features

SNU-COMPASS is designed with a set of robust features that cater to the diverse needs of both faculty and students, revolutionizing the academic planning experience at our university.

1. **Progress Tracking for Teachers**:
Faculty members can efficiently track students' academic progress. This feature provides a comprehensive overview of the courses completed, grades obtained, and overall academic performance, enabling teachers to offer timely guidance and support.

2. **Dynamic Addition of Minor Courses:**
Teachers have the flexibility to dynamically add new courses, such as SWAYAM, to the application. These added courses are seamlessly integrated into the system, counting towards a student's minor completion. This eliminates the need for extensive email communication with students regarding course updates, ensuring a real-time and efficient process.

3. **Personalized Progress Dashboard for Minor Pursuers**:
Students pursuing a minor can access a personalized progress dashboard. This feature empowers students to visualize their academic journey, providing insights into completed and upcoming minor-related courses. The dashboard aids in effective schedule planning, ensuring students stay on track with their minor requirements.

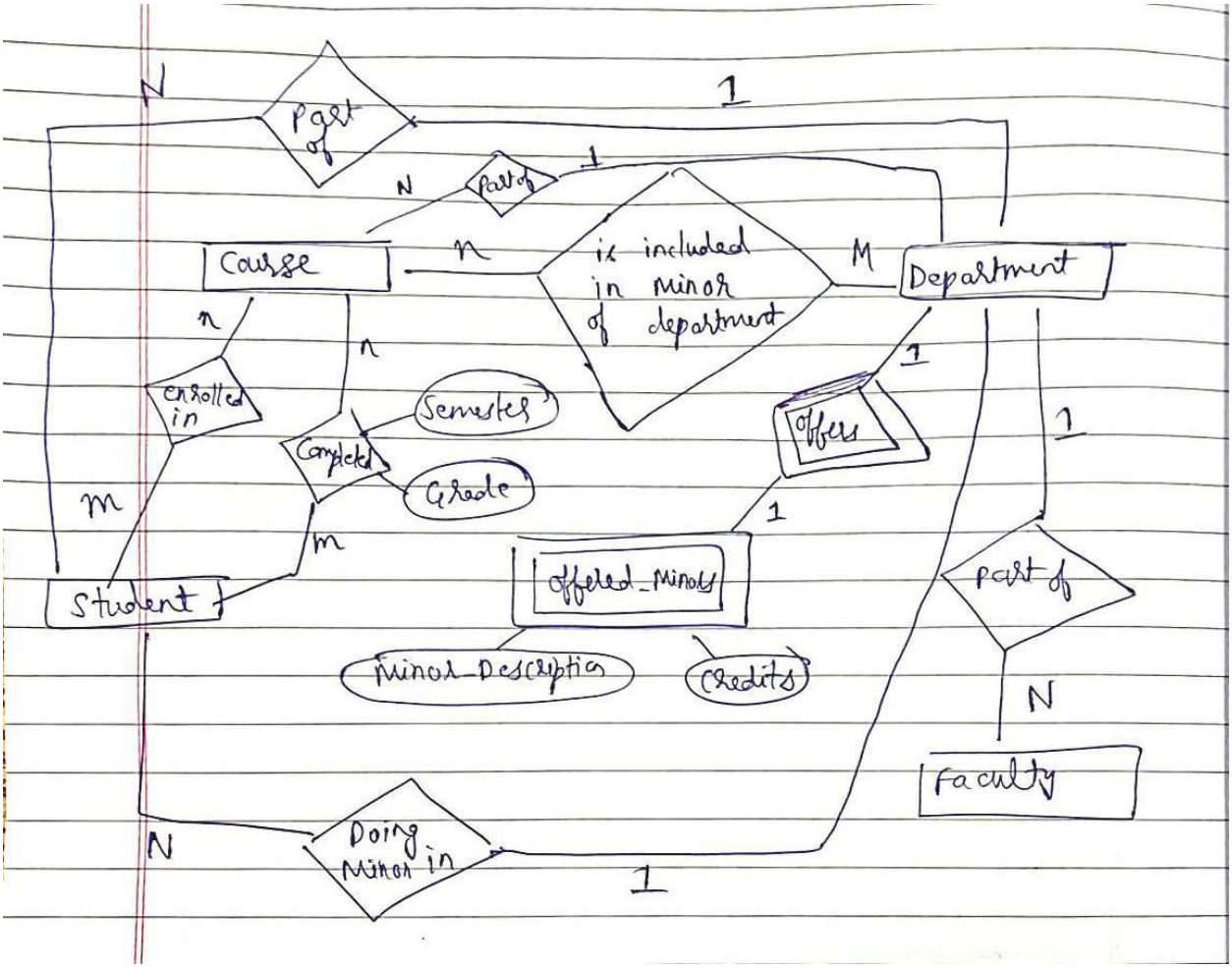
4. **Comprehensive Course Completion Overview for Non-Minor Students**:
For students not currently pursuing a minor, the application offers a comprehensive view of course completion across all departments. This feature serves as a valuable tool for students to assess their academic progress and consider potential minor options in various departments based on their existing coursework.

5. **User-Friendly Interface**:
SNU-COMPASS boasts a user-friendly interface, ensuring ease of navigation for both faculty and students. The intuitive design facilitates seamless interaction with the application, enhancing the overall user experience.

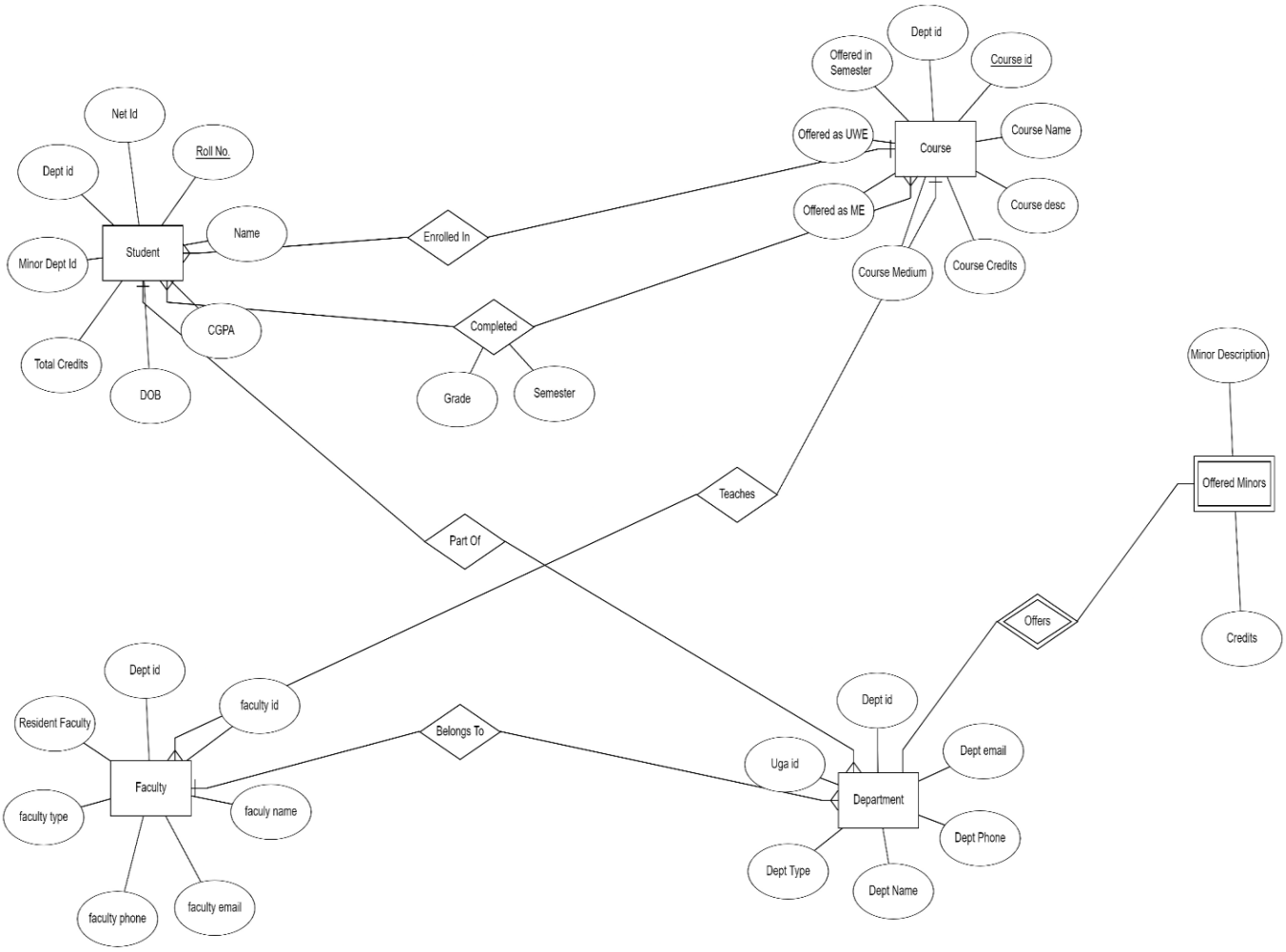
6. **Detailed Course Information**:
The application provides detailed information about each course, including course name, description, credits, and the department offering the course. This information aids students in making informed decisions about their academic paths.

By integrating these features, SNU-COMPASS emerges as a comprehensive and innovative solution, simplifying academic planning for both faculty and students while fostering a more dynamic and responsive educational environment at Shiv Nadar University.

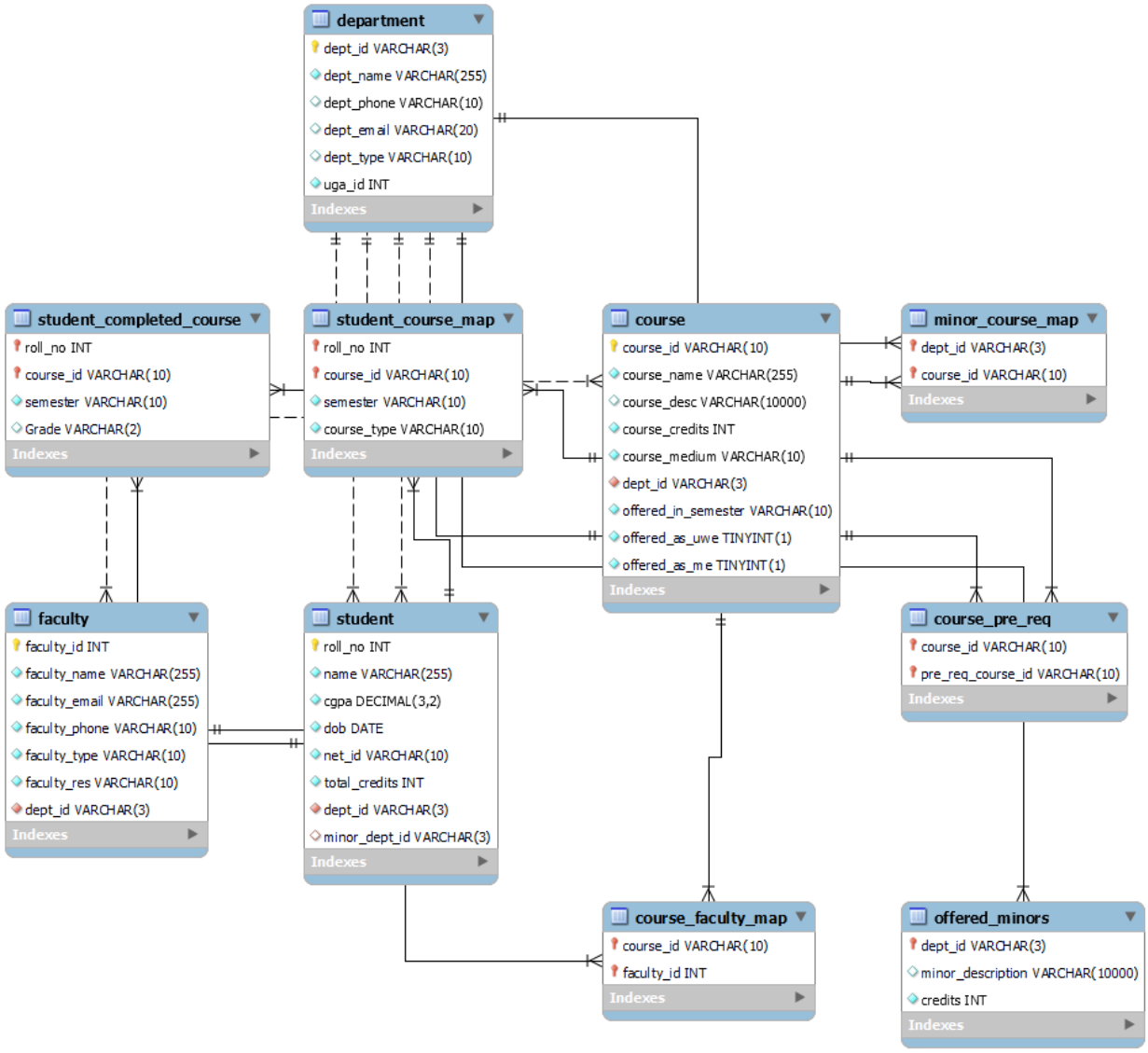
ER Model



(Hand drawn ER diagram, following the pattern taught in class)



(ER Diagram made in draw.io)



(ER diagram displaying proper information about database schema)

Relational Model

Department

dept_id	dept_name	dept_phone	dept_email	dept_type	uga_id

Student

roll_no	name	cgpa	dob	net_id	total_credits	dept_id	minor_dept_id

Course

course_id	course_name	course_desc	course_credits	course_medium	dept_id	offered_in_semester	offered_as_uwe	offered_as_me

Faculty

faculty_id	faculty_name	faculty_email	faculty_phone	faculty_type	faculty_res	dept_id

Student_Course_Map

roll_no	course_id	semester	course_type

Student_Completed_Course

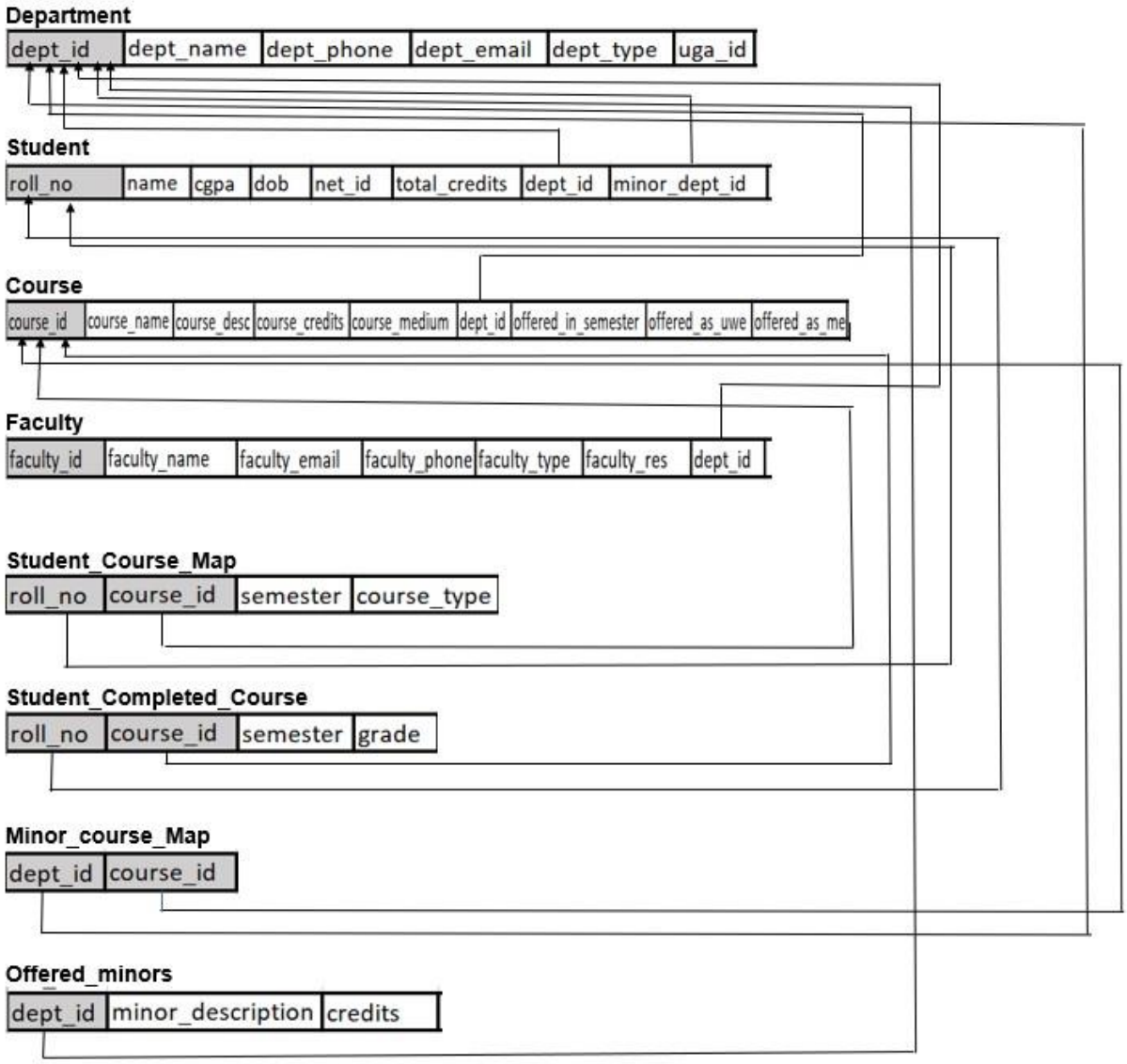
roll_no	course_id	semester	grade

Minor_course_Map

dept_id	course_id

Offered_minors

dept_id	minor_description	credits


Department
dept_id
dept_name
dept_phone
dept_email
dept_type
uga_id
Student
roll_no
name
cgpa
dob
net_id
total_credits
dept_id
minor_dept_id
Course
course_id
course_name
course_desc
course_credits
course_medium
dept_id
offered_in_semester
offered_as_uwe
offered_as_me
Faculty
faculty_id
faculty_name
faculty_email
faculty_phone
faculty_type
faculty_res
dept_id
Student_Course_Map
roll_no
course_id
semester
course_type
Student_Completed_Course
roll_no
course_id
semester
grade
Minor_course_Map
dept_id
course_id
Offered_minors
dept_id
minor_description
credits

Normalization Relations

1) Department

dept_id	dept_name	dept_phone	dept_email	dept_type	uga_id

Here, **dept_id** is the primary key and it determines all the other non-prime attributes.

Functional Dependencies:

1. dept_id -> dept_name
2. dept_id -> dept_phone
3. dept_id -> dept_email
4. dept_id -> dept_type
5. dept_id -> uga_id

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

2) Student

roll_no	name	cgpa	dob	net_id	total_credits	dept_id	minor_dept_id

Here, **roll_no** is the primary key and it determines all the other non-prime attributes.

Functional Dependencies:

1. roll_no -> name
2. roll_no -> cgpa
3. roll_no -> dob

4. roll_no -> net_id
5. roll_no -> total_credits
6. roll_no -> dept_id
7. roll_no -> minor_dept_id

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

3) Course

course_id	course_name	course_desc	course_credits	course_medium	dept_id	offered_in_semester	offered_as_uwe	offered_as_me

Here, **course_id** is the primary key and it determines all the other non-prime attributes.

Functional Dependencies:

1. course_id -> course_name
2. course_id -> course_desc
3. course_id -> course_credits
4. course_id -> course_medium
5. course_id -> dept_id
6. course_id -> offered_in_semester
7. course_id -> offered_as_uwe
8. course_id -> offered_as_me

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

4) Faculty

faculty_id	faculty_name	faculty_email	faculty_phone	faculty_type	faculty_res	dept_id

Here, **faculty_id** is the primary key and it determines all the other non-prime attributes.

Functional Dependencies:

1. faculty_id -> faculty_name
2. faculty_id -> faculty_email
3. faculty_id -> faculty_phone
4. faculty_id -> faculty_type
5. faculty_id -> faculty_res
6. faculty_id -> dept_id

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

5) Student_Course_Map

roll_no	course_id	semester	course_type

Here, **roll_no & course_id** together are the primary keys, and they determine all the other non-prime attributes.

Functional Dependencies:

1. (roll_no, course_id) -> semester
2. (roll_no, course_id) -> course_type

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

6) Student_Completed_Course

roll_no	course_id	semester	grade

Here, **roll_no & course_id** together are the primary keys, and they determine all the other non-prime attributes.

Functional Dependencies:
1. (roll_no, course_id) -> semester
2. (roll_no, course_id) -> grade

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

7) Minor_course_map

dept_id	course_id

Here, **dept_id & course_id** together are the primary keys.

The relation is in:

1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

8) Offered_minors

dept_id	minor_description	credits

Here, **dept_id** is the primary key.

Functional Dependencies:

1. dept_id -> minor_description
2. dept_id -> credits

The relation is in:
1NF as it has atomic attributes.
2NF as it has no partial dependency.
3NF as it has no transitive dependency.

Hence, the relation is in **3NF**.

Frontend Screenshots & Backend Queries

1) GetCompletedCourse

```
if (req.method === 'GET') {
const { rollNo,minorDep} = req.query;
  const results = await new Promise((resolve, reject) => {
    db.query(`select course.course_id,course_name,semester from Student_Completed_Course
    join course on course.course_id=Student_Completed_Course.course_id
    join student on student.roll_no=Student_Completed_Course.roll_no
     where student.roll_no=${rollNo} and left(course.course_id,3)=${minorDep};
    `, (error, results) => {
      if (error) {
        reject(error);
      } else {
        resolve(results);
      }
    });
```

```
const response4 = await fetch(`https://snucompass.vercel.app/api/GetCompletedCourses?rollNo=${rollNumber}&minorDep="$
{studentData[0].dept_id}"`);
if (!response4.ok) {
  throw new Error(`Error fetching data: ${response2.statusText}`);
}
```

Completed Courses

Course ID	Course Name	Semester
CSD101	Introduction to Computing and Programming	M2023
CSD102	Data Structures	M2023
CSD204	Operating Systems	M2023
CSD311	Artificial Intelligence	M2023

2) GetCourseList

```
if (req.method === 'GET') {
  const results = await new Promise((resolve, reject) => {
    db.query('SELECT * FROM course', (error, results) => {
      if (error) {
        reject(error);
      } else {
        resolve(results);
      }
    });
  });
```

3) GetEnrolledMinorCourses

```
try {
  await middleware(req, res);
  if (req.method === 'GET') {
  const { dept, rollNo } = req.query;
  console.log(dept)
  console.log(rollNo)
    const results = await new Promise((resolve, reject) => {
      db.query(`select t1.course_id,course.course_name from course
      Join (select course_id from Student_Course_Map where left(course_id,3)=${dept} and roll_no=${rollNo}) as t1
      on course.course_id=t1.course_id;`, (error, results) => {
        if (error) {
          reject(error);
        } else {
          resolve(results);
        }
      });
    });
```

```
const response2 = await fetch(`https://snucompass.vercel.app/api/GetEnrolledMinorCourses?dept="${studentData[0].dept_id}"&
rollNo=${rollNumber}`);
if (!response2.ok) {
  throw new Error(`Error fetching data: ${response2.statusText}`);      You, 7 minutes ago • Uncommitted changes
}
```

User Dashboard

Welcome Vijay Varma

Pursuing Minor in Computer Science Progress:

84.21%

Completed Courses

Course ID	Course Name	Semester
CSD101	Introduction to Computing and Programming	M2023
CSD102	Data Structures	M2023
CSD204	Operating Systems	M2023
CSD311	Artificial Intelligence	M2023

Ongoing Courses

Course ID	Course Name
CSD205	Discrete Mathematics

User Dashboard

Welcome Hrithik Roshan

You are not pursuing a minor in any department.

Your Progress in Minor Departments

Management

Credits: 18

The Minor in Management offers undergraduates exposure to key management concepts for better job readiness. It benefits diverse career paths and provides a competitive edge for those considering an MBA.

Read More | Your Progress

Communication

Credits: 18

The Minor in Communication offered by the Department of Communication is structured in a way that takes the student on the journey a story takes till it manifests itself on screen; from Ideation and Writing, Basic Photography and Sound

Read More | Your Progress

Computer Science

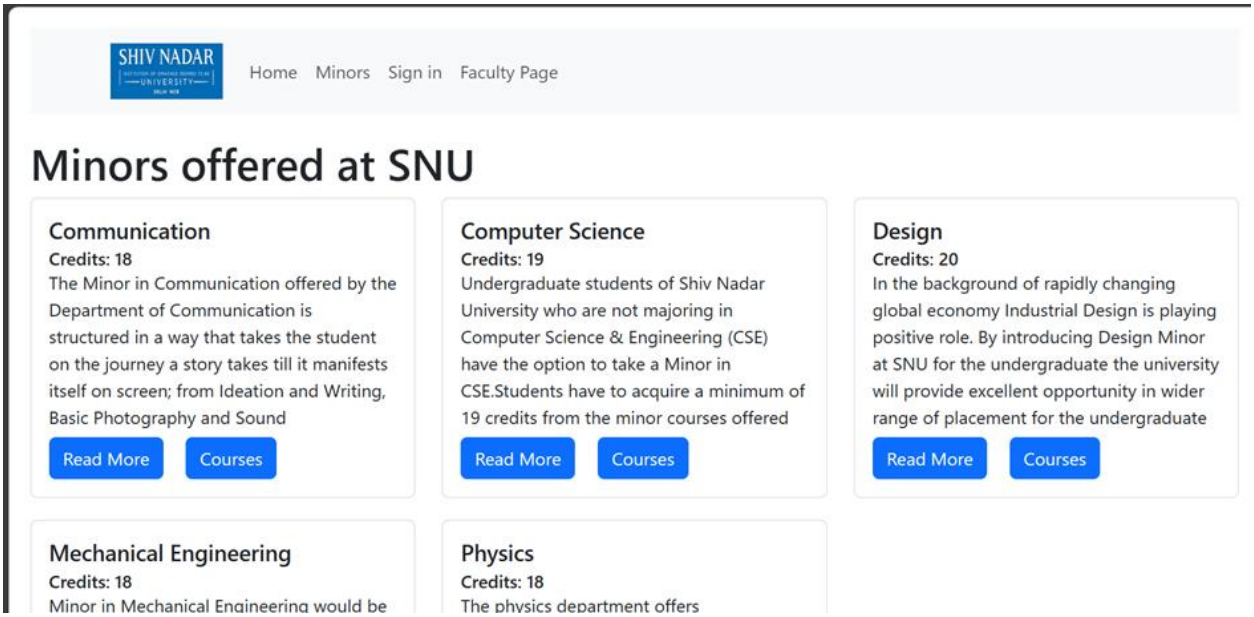
Credits: 19

Undergraduate students of Shiv Nadar University who are not majoring in Computer Science & Engineering (CSE) have the option to take a Minor in CSE.Students have to acquire a minimum of 19 credits from the minor courses offered

Read More | Your Progress

4) GetMinorDep

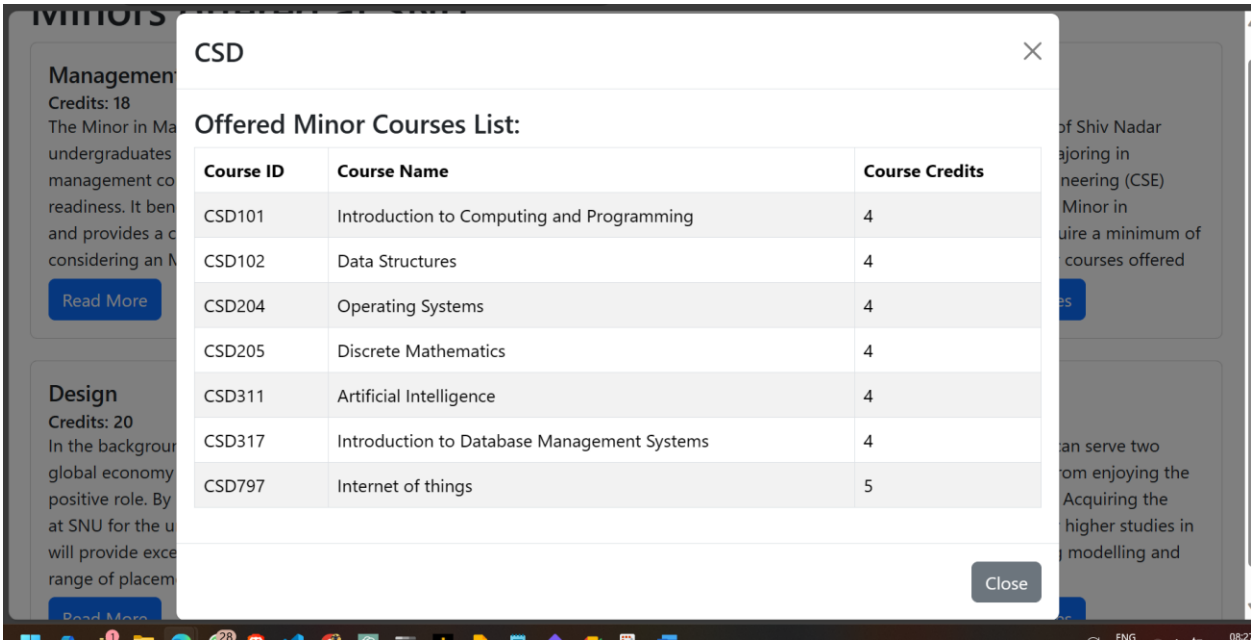
```
try {
  const response = await fetch('https://snucompass.vercel.app/api/GetMinorDep');
  if (!response.ok) {
    throw new Error(`Error fetching data: ${response.statusText}`);
  }
```



5) GetMinorCourseMap

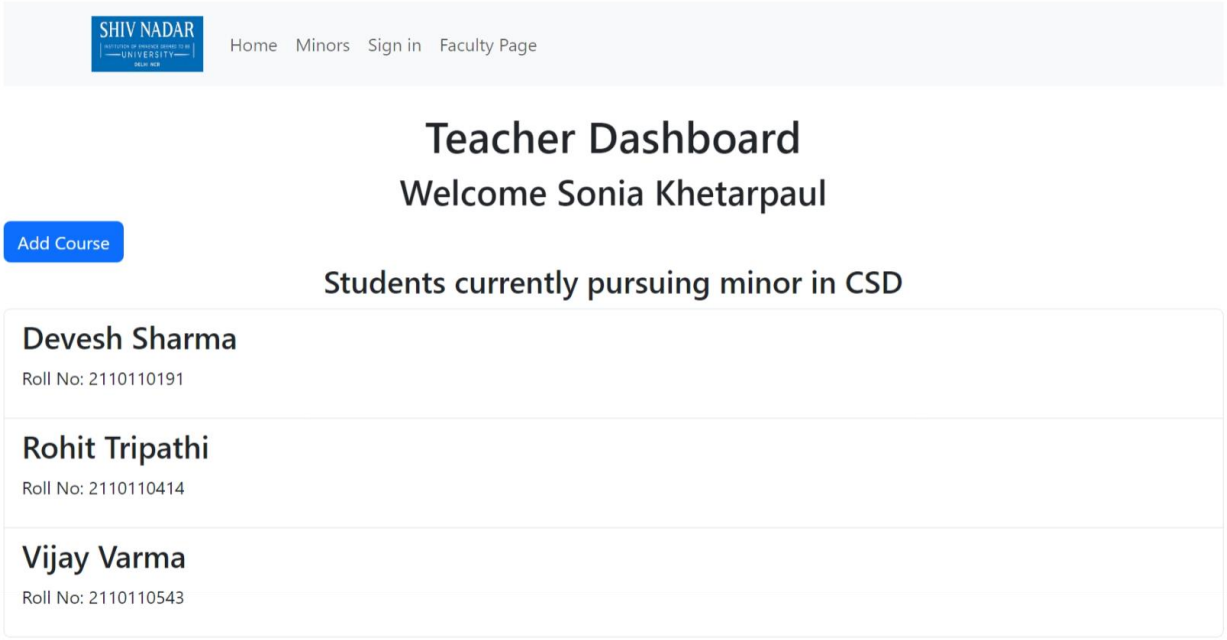
```
try {
  if (req.method === 'GET') {
    const { dept } = req.query;
    const results = await new Promise((resolve,reject) => {
      db.query(`select course.course_id,course.course_name,course.course_credits from course
      join Minor_course_Map on Minor_course_Map.dept_id=course.dept_id where course.dept_id=${dept}
      and course.course_id=Minor_course_Map.course_id`,(error,results) => {
        if (error) {
          reject(error);      Rahuldj, 4 days ago • Fixed minor dashboard modal and all endpoints
        } else {
          resolve(results);
        }
      });
    });
```

```
const response = await fetch(`https://snucompass.vercel.app/api/GetMinorCourseMap?dept="${minor}"`);
if (!response.ok) {
  throw new Error(`Error fetching data: ${response.statusText}`);
}
```



6) GetMinorStudents

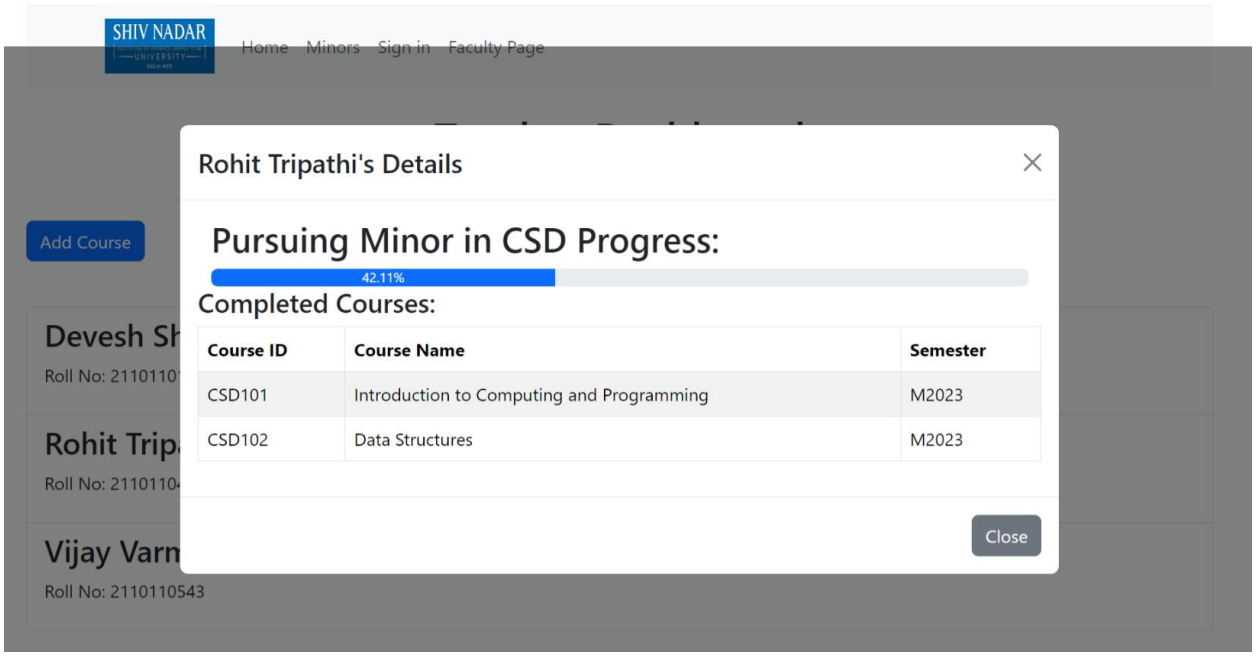
```
export default async function(req, res) {
  try {
    const{fac_email}=req.query;
    const email=fac_email+"@snu.edu.in";
    console.log(email)
    if (req.method === 'GET') {
      const results = await new Promise((resolve, reject) => {
        db.query(`select faculty_name,faculty.dept_id,roll_no,name
        from faculty join student on faculty.dept_id=student.minor_dept_id
        where faculty.dept_id=(select dept_id from faculty
            where faculty_email="${email}");`, (error, results) => {
          if (error) {
            reject(error);
          } else {
            resolve(results);
          }
        });
      });
```



7) GetProgressPercent

```
export default async function (req,res) {
  await middleware(req,res);
  try {
    if (req.method === 'GET') {
      const { rollNo,minorDep } = req.query;
      const results = await new Promise((resolve,reject) => {
        db.query(`select sum(course_credits)/(SELECT credits from offered_minors where dept_id=${minorDep})*100 as percentage from
        Student_Completed_Course
                join course on course.course_id=Student_Completed_Course.course_id
                join student on student.roll_no=Student_Completed_Course.roll_no
              where student.roll_no=${rollNo} and left(course.course_id,3)=${minorDep};
        `,(error,results) => {
          if (error) {
            reject(error);       Rahuldj, 4 days ago • Add new usernames and passwords to SignIn page an…
          } else {
            resolve(results);
          }
        });
```

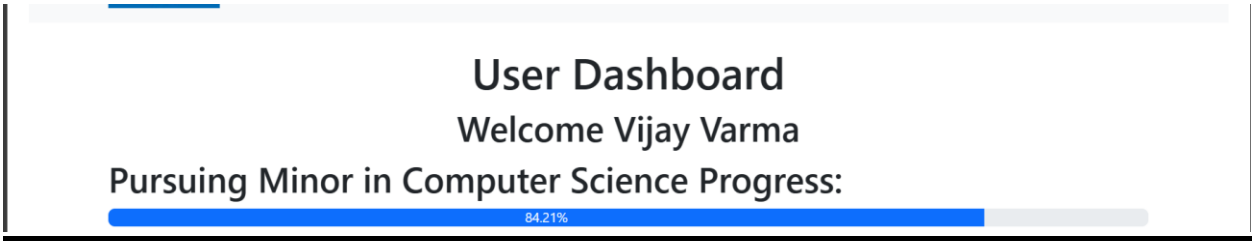
```
const fetchCompletedCourses = async () => {
  try {
    const response = await fetch(`https://snucompass.vercel.app/api/GetCompletedCourses?rollNo=${student?.roll_no}&
    minorDep="${student?.dept_id}"`);
    if (!response.ok) {
      throw new Error(`Error fetching data: ${response.statusText}`);
    }
    const coursesData = await response.json();
    setCompletedCourses(coursesData);


  const response3 = await fetch(`https://snucompass.vercel.app/api/GetProgressPercent?rollNo=${student?.roll_no}&minorDep="$
  {student?.dept_id}"`);
  if (!response3.ok) {
      throw new Error(`Error fetching data: ${response2.statusText}`);
  }
```



8) GetStudentData

```
export default async function(req, res) {
  await middleware(req, res);
  try {
    if (req.method === 'GET') {
    const { rollNo } = req.query;
      const results = await new Promise((resolve, reject) => {
        db.query(`SELECT name,department.dept_id,department.dept_name from student
        LEFT JOIN department on department.dept_id=student.minor_dept_id
        WHERE student.roll_no=${rollNo}
        `, (error, results) => {
          if (error) {
            reject(error);
          } else {
            resolve(results);
          }
        });
      });
      console.log(results)
      res.status(200).json(results);
    } else {
      res.status(405).json({ error: 'Method not allowed' });
    }
  } catch (error) {
```

```
try {
  const response = await fetch(`https://snucompass.vercel.app/api/GetStudentData?rollNo=${rollNumber}`);
  if (!response.ok) {
    throw new Error(`Error fetching data: ${response.statusText}`);
  }
  const studentData = await response.json();
  console.log(studentData[0]);
  if (studentData[0].dept_id === null) {
```



Limitations

While SNU-COMPASS strives to provide an enhanced academic planning experience, there are certain limitations to the current implementation, primarily due to time constraints during the development phase. It is important to acknowledge these limitations for a transparent understanding of the system's scope.

1. **Consolidated View for Specializations**:
Initially, we planned to include a consolidated view for specializations offered across various departments. Unfortunately, due to time constraints, this feature could not be implemented in the current version. We recognize the significance of providing a holistic view of specializations, and while not realized in the current release, a similar model to the one described for minors can be adopted in future iterations.

2. **Limited Integration with External Systems**:
Integrating the application with external systems, such as the university's student information system or course registration system, could enhance the accuracy and efficiency of data updates. However, due to time constraints, this level of integration was not achieved in the current version.

3. **Enhanced Communication Features**:
While the application streamlines information access, it does not currently facilitate direct communication between faculty and students. Integrating enhanced communication features, such as direct messaging or discussion forums, was not within the scope of the initial release.

It is essential to recognize these limitations as areas for potential improvement and expansion in future iterations. Despite these constraints, SNU-COMPASS remains a valuable tool for simplifying academic planning and fostering informed decision-making within the SNU academic community.

Novelty

In the academic landscape at Shiv Nadar University (SNU), students often encounter challenges in accessing accurate and current information about available minors. Traditionally, students have navigated through outdated PDF documents, resulting in a laborious process of seeking guidance through multiple email interactions with the Undergraduate Advisor (UGA).

Our project, SNU-COMPASS, pioneers a transformative solution to this long-standing issue by establishing a centralized platform that serves as a comprehensive resource for information related to minors offered by various departments at SNU. The primary novelty lies in the redefinition of a cumbersome process into a streamlined and user-friendly experience.

Key Novelty Features:

1. **One-Stop Destination**: SNU-COMPASS acts as a singular point of reference for students seeking information on available minors. Instead of navigating through multiple channels and dealing with outdated documents, students can now access a centralized and up-to-date platform.

2. **Detailed Minor Descriptions**: The application offers comprehensive explanations of minors, including course requirements, credit details, and any additional information provided by the respective departments. This surpasses the limitations of traditional documents, providing a richer understanding of the academic offerings.

3. **UGA Interaction Simplification**: Recognizing the historical reliance on email communication with the Undergraduate Advisor (UGA), our application simplifies this process. Students can now efficiently gather information about minors without the need for extensive email exchanges.

4. **Dual Functionality for Faculty and Students**: Our application recognizes the diverse needs of both faculty and students. Faculty members can use the system to manage and update minor information, contributing to a dynamic and responsive database.

By addressing the limitations of current information dissemination methods and enhancing the accessibility and clarity of minor-related data, SNU-COMPASS stands as an innovative solution, simplifying academic planning for students and promoting efficiency within the academic community at Shiv Nadar University.

References

1. **React Bootstrap**:
The development of the user interface in SNU-COMPASS utilized React Bootstrap, a popular front-end framework that integrates Bootstrap components seamlessly with React.
Reference: React Bootstrap Documentation

2. **Next.js Documentation**:
Next.js played a pivotal role in building the web application, providing a robust framework for React-based server-side rendering and routing.
Reference: Next.js Documentation

3. **MySQL Documentation (W3Schools)**:
The project leveraged MySQL for the database management system. The W3Schools documentation for MySQL served as a valuable resource for understanding and implementing database queries and interactions.
Reference: W3Schools MySQL Documentation