SOYAMSIDH DAS

EDTECH PROJECT

In [15]:
```python
# Step 1: User Class
class User:
    def __init__(self, user_id, name, email, password):
        self.user_id = user_id
        self.name = name
        self.email = email
        self.password = password

    def update_email(self, new_email):
        self.email = new_email

    def update_password(self, new_password):
        self.password = new_password

    def validate_credentials(self, email, password):
        return self.email == email and self.password == password
```

In [17]:
```python
# Step 2: Learner Class
class Learner(User):
    def __init__(self, user_id, name, email, password):
        super().__init__(user_id, name, email, password)
        self.courses = []

    def enroll_course(self, course):
        if course not in self.courses:
            self.courses.append(course)
            course.add_learner(self)

    def drop_course(self, course):
        if course in self.courses:
            self.courses.remove(course)
            course.remove_learner(self)
```

In [18]:
```python
# Step 3: Instructor Class
class Instructor(User):
    def __init__(self, user_id, name, email, password):
        super().__init__(user_id, name, email, password)
        self.courses_taught = []

    def add_course(self, course):
        if course not in self.courses_taught:
            self.courses_taught.append(course)

    def remove_course(self, course):
        if course in self.courses_taught:
            self.courses_taught.remove(course)
```

In [19]:
```python
# Step 4: Course Class
class Course:
    def __init__(self, course_id, title, instructor):
        self.course_id = course_id
        self.title = title
        self.instructor = instructor
        self.learners = []

    def add_learner(self, learner):
        if learner not in self.learners:
            self.learners.append(learner)

    def remove_learner(self, learner):
        if learner in self.learners:
            self.learners.remove(learner)

    def list_learners(self):
        return [learner.name for learner in self.learners]
```

In [20]:
```python
# Step 5: Enrollment Class
class Enrollment:
    def __init__(self):
        self.enrollments = []

    def enroll(self, learner, course):
        learner.enroll_course(course)
        self.enrollments.append((learner, course))

    def drop(self, learner, course):
```

```
        learner.drop_course(course)
        updated_enrollments = []
        for l, c in self.enrollments:
            if l != learner or c != course:
                updated_enrollments.append((l, c))
        self.enrollments = updated_enrollments
```

In [21]:

```
# Step 6: SLTechBackend Class
class SLTechBackend:
    def __init__(self):
        self.users = []
        self.courses = []
        self.enrollment_system = Enrollment()

    def add_user(self, user):
        self.users.append(user)

    def add_course(self, course):
        self.courses.append(course)

    def enroll_learner(self, learner_id, course_id):
        learner = next((user for user in self.users if isinstance(user, Learner) and user.user_id == learner_id
        course = next((c for c in self.courses if c.course_id == course_id), None)
        if learner and course:
            self.enrollment_system.enroll(learner, course)
            print(f"{learner.name} enrolled in {course.title}")
        else:
            print("Enrollment failed.")

    def drop_learner(self, learner_id, course_id):
        learner = next((user for user in self.users if isinstance(user, Learner) and user.user_id == learner_id
        course = next((c for c in self.courses if c.course_id == course_id), None)
        if learner and course:
            self.enrollment_system.drop(learner, course)
            print(f"{learner.name} dropped {course.title}")
        else:
            print("Dropping course failed.")

    def list_enrolled_learners(self, course_id):
        course = next((c for c in self.courses if c.course_id == course_id), None)
        if course:
            learners = course.list_learners()
            return learners
        return []

    def get_user_info(self, user_id):
        user = next((u for u in self.users if u.user_id == user_id), None)
        if user:
            print(f"User ID: {user.user_id}")
            print(f"Name: {user.name}")
            print(f"Email: {user.email}")
        else:
            print("User not found.")

    def manage_user_input(self):
        while True:
            print("\n--- EdTech Backend System ---")
            print("1. Add User")
            print("2. Add Course")
            print("3. Enroll Learner")
            print("4. Drop Learner")
            print("5. List Enrolled Learners")
            print("6. Get User Information")
            print("7. Exit")
            choice = input("Enter your choice: ")

            if choice == '1':
                user_type = input("Enter user type (learner/instructor): ").lower()
                user_id = input("Enter user ID: ")
                name = input("Enter name: ")
                email = input("Enter email: ")
                password = input("Enter password: ")

                if user_type == 'learner':
                    self.add_user(Learner(user_id, name, email, password))
                    print(f"Learner {name} added.")
                elif user_type == 'instructor':
                    self.add_user(Instructor(user_id, name, email, password))
                    print(f"Instructor {name} added.")
                else:
                    print("Invalid user type.")

            elif choice == '2':
```

```
                course_id = input("Enter course ID: ")
                title = input("Enter course title: ")
                instructor_id = input("Enter instructor ID: ")
                instructor = next((user for user in self.users if isinstance(user, Instructor) and user.user_id

                if instructor:
                    course = Course(course_id, title, instructor)
                    self.add_course(course)
                    instructor.add_course(course)
                    print(f"Course {title} added.")
                else:
                    print("Instructor not found.")

            elif choice == '3':
                learner_id = input("Enter learner ID: ")
                course_id = input("Enter course ID: ")
                self.enroll_learner(learner_id, course_id)

            elif choice == '4':
                learner_id = input("Enter learner ID: ")
                course_id = input("Enter course ID: ")
                self.drop_learner(learner_id, course_id)

            elif choice == '5':
                course_id = input("Enter course ID: ")
                learners = self.list_enrolled_learners(course_id)
                if learners:
                    print("Enrolled Learners: ", learners)
                else:
                    print("No learners found.")

            elif choice == '6':
                user_id = input("Enter user ID: ")
                self.get_user_info(user_id)

            elif choice == '7':
                break

            else:
                print("Invalid choice. Try again.")
```

In [22]:
```
# Driver Code

if __name__ == "__main__":
    sltech_backend = SLTechBackend()
    sltech_backend.manage_user_input()
```

```
--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
User not found.

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
Learner Ramesh added.

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
```

```
Instructor not found.

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners

6. Get User Information
7. Exit
Instructor Dr. Soyam added.

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
Course Python programming added.

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
Ramesh enrolled in Python programming

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
Enrolled Learners:  ['Ramesh']

--- EdTech Backend System ---
1. Add User
2. Add Course
3. Enroll Learner
4. Drop Learner
5. List Enrolled Learners
6. Get User Information
7. Exit
```