Student Registration System - Design Document

Executive summary

A web-based registration system allowing students to register, view schedule, update personal info; instructors view roster and enter grades; admins manage courses/terms.

Scope & objectives

- Student registration & course enrollment
- Class schedule viewing
- Personal information update
- Instructor roster view & grade entry
- Admin course/term management
- Audit logs & reporting

Stakeholders & actors

- Students (primary users)
- Instructors (grade entry, rosters)
- Registrar / Admin (course setup, approvals)
- System Administrator (deployment, backup)
- External systems (authentication/SSO, payment gateway if fees)

Requirements

Functional

1. Student login/logout (SSO optional).
2. Browse course catalog, search/filter.
3. Register/drop courses (with pre-req checks, seat limits).
4. View weekly schedule and printable timetable.
5. Update personal contact info.
6. Instructor: view class roster, record grades, submit final grades.
7. Admin: create/modify courses, sections, term dates.
8. Notifications (email/sms) on registration and seat changes.
9. Audit/history of changes.

Non-functional (sample)

1. Availability: 99.5% during registration windows.
2. Security: role-based access, TLS, input validation.
3. Performance: page load < 2s for listing up to 200 courses.
4. Scalability: support 50k students, 1k concurrent registrations.
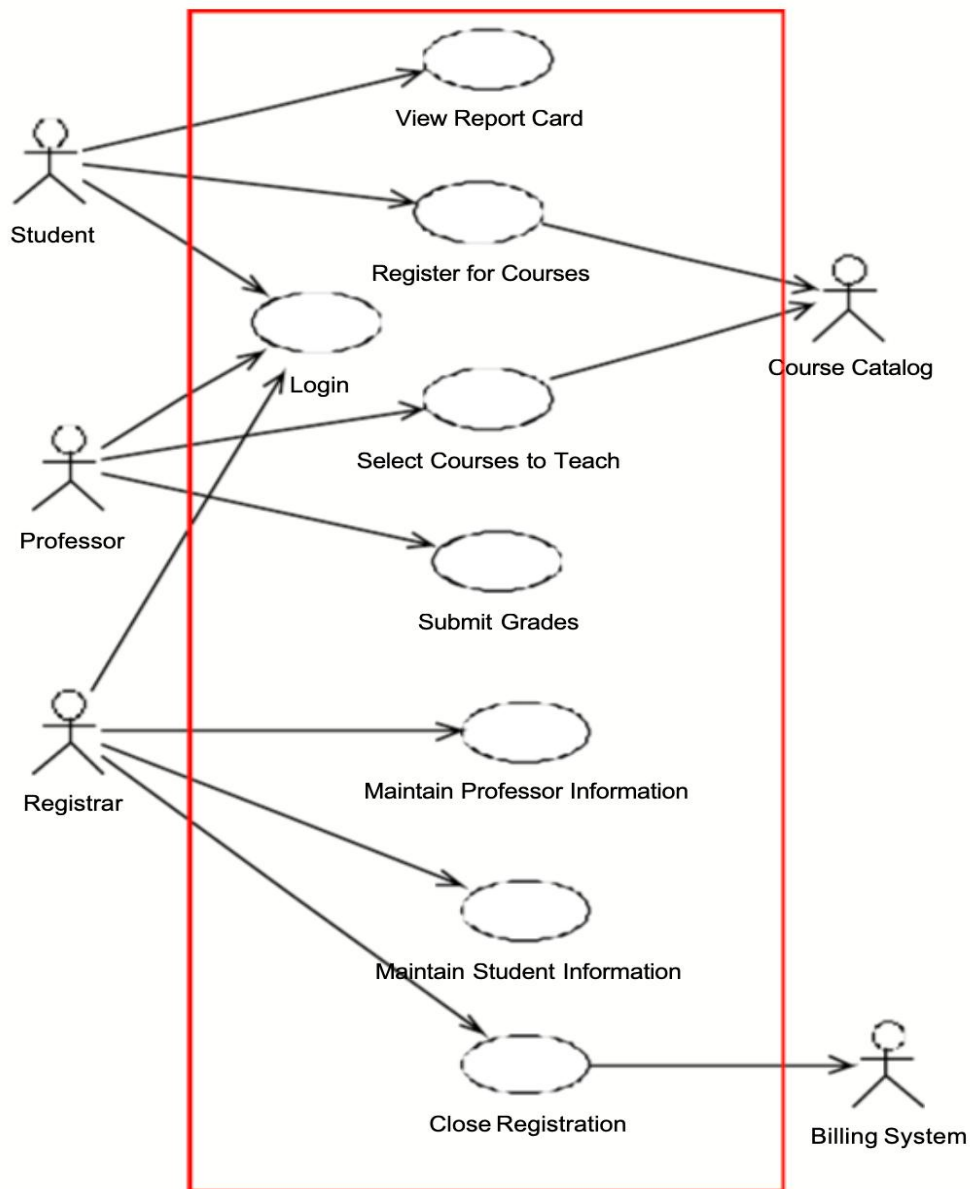5. Accessibility: WCAG AA compliance.

Architecture overview

- High-level tiered architecture:
- Presentation layer (web frontend)
- Application layer (business logic, services)
- Data layer (RDBMS — PostgreSQL/MySQL)
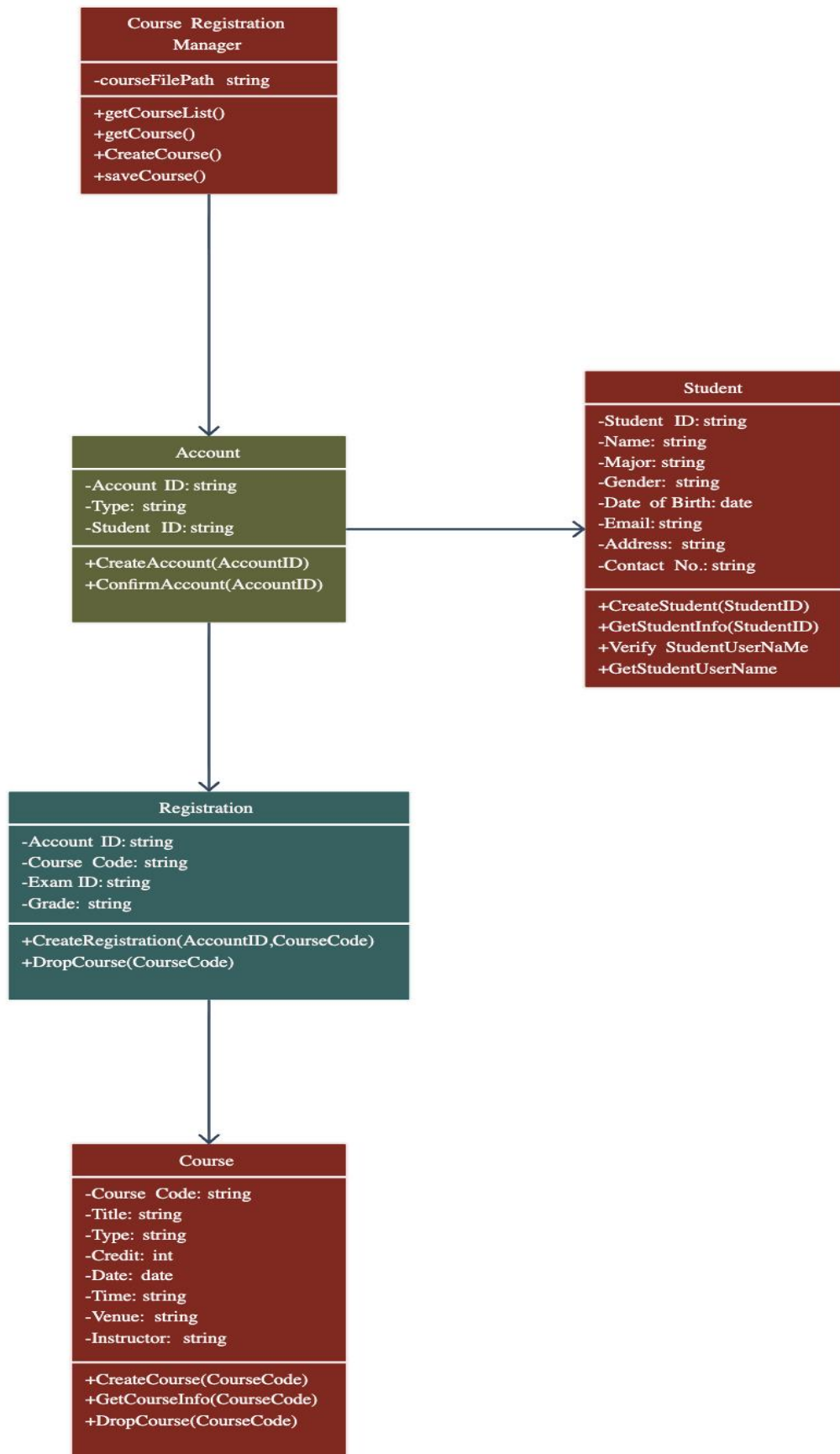- External integration (SSO LDAP/SAML, email service)
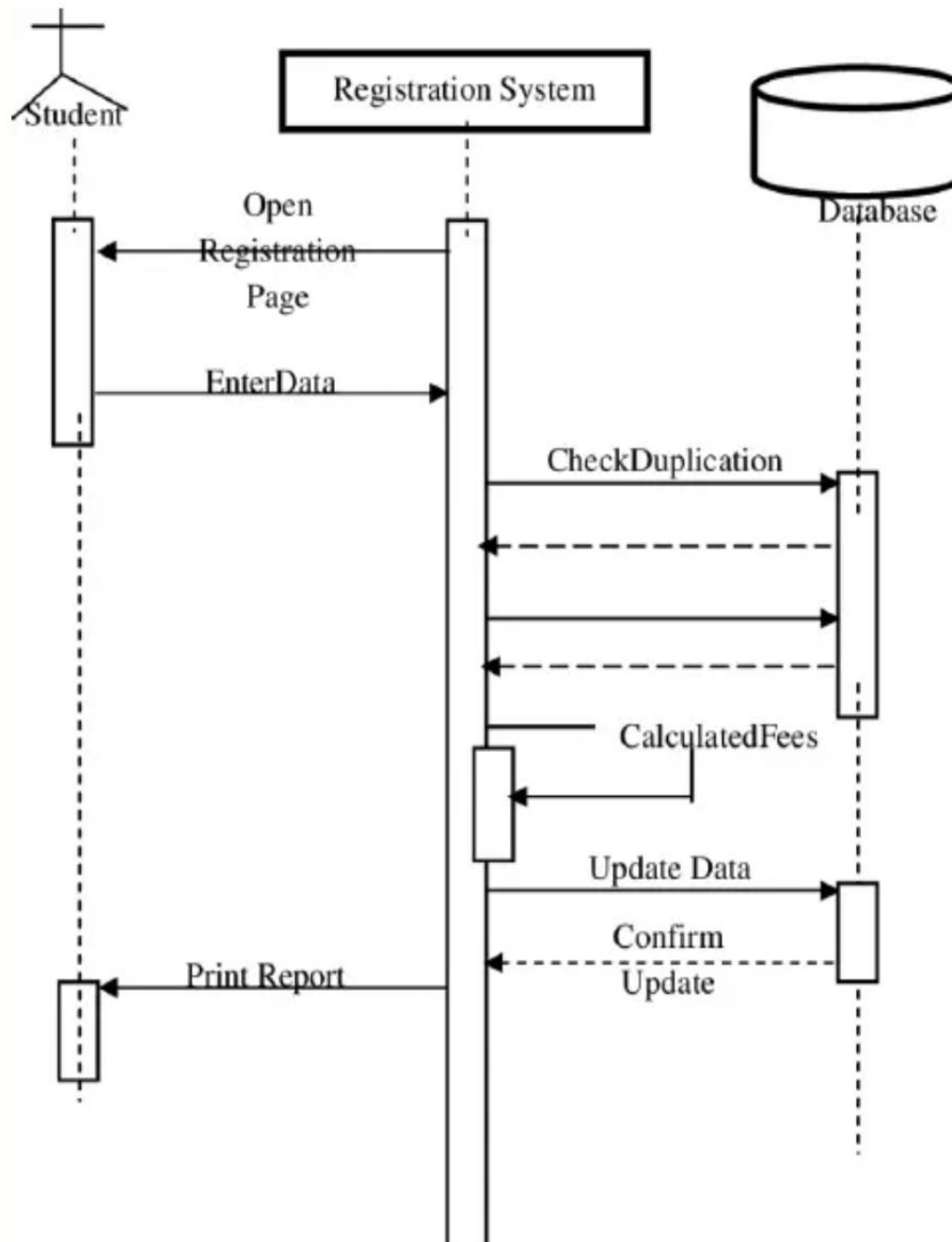
Object-Oriented Design

Use Case Model

Actors: Student, Instructor, Admin, System. Use cases: Login, Browse Courses, Register Course, Drop Course, View Schedule, Update Profile, Enter Grades, View Class Roster, Manage Courses.
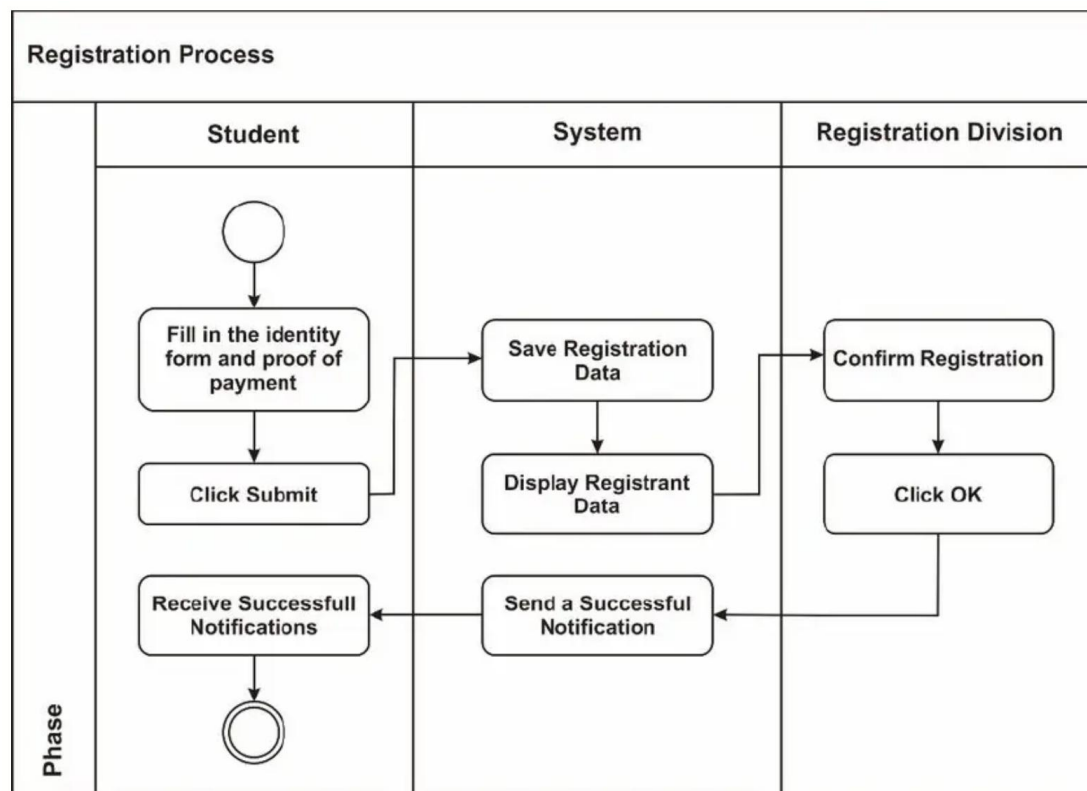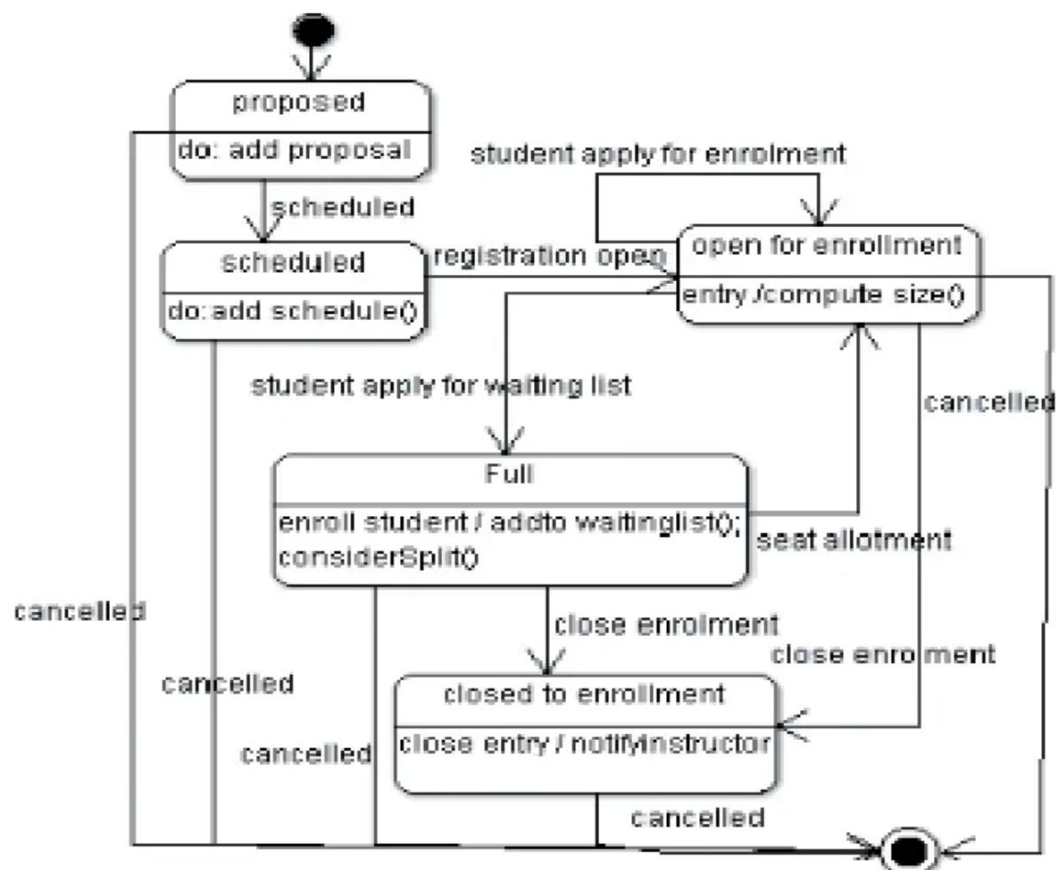
Class Diagram

Course Registration Manager

-courseFilePath string

+getCourseList()
+getCourse()
+CreateCourse()
+saveCourse()

Account

-Account ID: string
-Type: string
-Student ID: string

+CreateAccount(AccountID)
+ConfirmAccount(AccountID)

Student

-Student ID: string
-Name: string
-Major: string
-Gender: string
-Date of Birth: date
-Email: string
-Address: string
-Contact No.: string

+CreateStudent(StudentID)
+GetStudentInfo(StudentID)
+Verify StudentUserNaMe
+GetStudentUserName

Registration

-Account ID: string
-Course Code: string
-Exam ID: string
-Grade: string

+CreateRegistration(AccountID,CourseCode)
+DropCourse(CourseCode)

Course

-Course Code: string
-Title: string
-Type: string
-Credit: int
-Date: date
-Time: string
-Venue: string
-Instructor: string

+CreateCourse(CourseCode)
+GetCourseInfo(CourseCode)
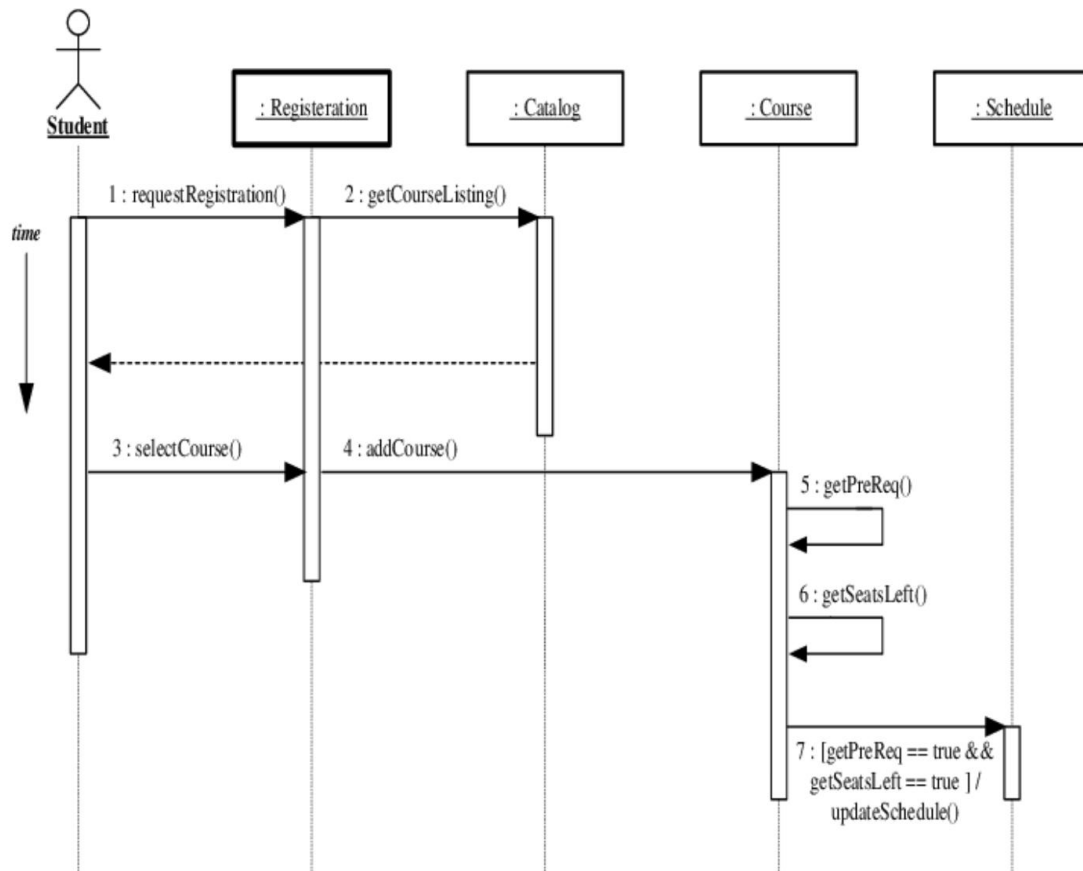+DropCourse(CourseCode)

Sequence Diagrams

Activity Diagram
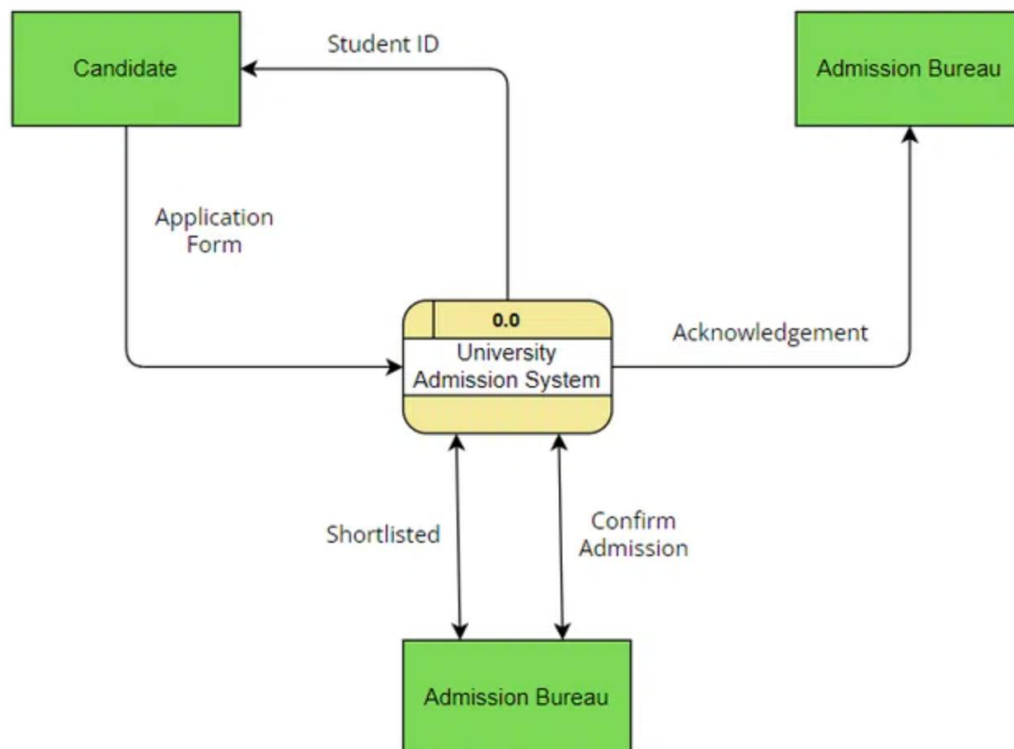

State Chart Diagram

Collaboration Diagrams

Procedure-Oriented Design

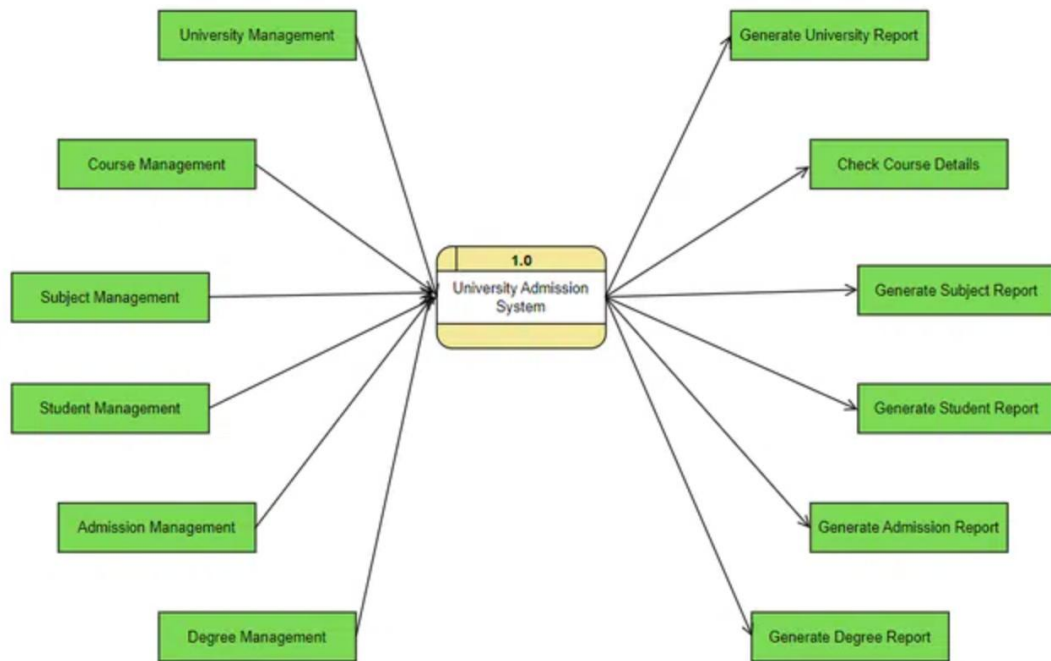## Data Flow Diagrams (DFD)

DFD Level 0 (Context diagram):

System: Student Registration System

External Entities: Candidate , Admission Bureau , Selection Committee

DFD Level 1 (Major processes):

- Generating University Report
- Generating Student Report
- Generating Subject Report
- Generating Admission Report
- Generating Degree Report
- Checking Course Details.

DFD Level 2 (Decompose Register/Drop):