**Lab 01**

**DMIT 2018 - A01**

**Purpose of the Report**

The purpose of this report is to introduce a new nait system that will be separated from the current nait system and the only person who can access this system is the admin. The NAIT Dental and Oral Surgery Clinic is open to Applicants who is interested on a dentistry business or career. Applicants can go to a different designation that will interest them and improve their skills before they graduate and to make them become in demand in the job market. Patients can register and book their plan procedure. Each procedure can be book on class hours and students who gained the right experience and practice will conduct the procedure. The system and the admin is responsible for CRUD of its data. Each entity on this report has its own responsibility that helps the system work properly. Each diagram and cases in this report will show how entities of the business diagram connects to each other and show what it does.

**Purpose of the Project**

The purpose of this project is to implement a rudimentary business system for the NAIT Dental and Oral Surgery Clinic. The intent of the clinic is to provide an educational facility for dentists and oral surgeons separate from NAIT. The clinic will also provide low cost dental and oral surgical services to the public. The system will allow for the enrollment of students into courses offered by the clinic, scheduling of appointments for patients, as well as all other functions in between to allow for the initialization and management of these two main functions. The current system only allows for the administrator to access the database and interact with the server. However, the client wishes to allow other users, such as students and instructors, to have access to the database through a user friendly web interface in the future.

**System Requirements**

The required system needs to be able to allow the administrative staff at NAIT run a new department separately from the main campus. Administrators need to be able to create and schedule classes, manage staff, manage students, schedule appointments online, and track inventory. At launch people from the public will not have access to the system. A portal to assist with billing insurance companies would be an asset to the system. The payroll is handled by the NAIT system and is out of scope for this project.

**Stakeholder Analysis**

* Government
* NAIT
* Instructors
* Students
* Staff
* Patients
* Vendors

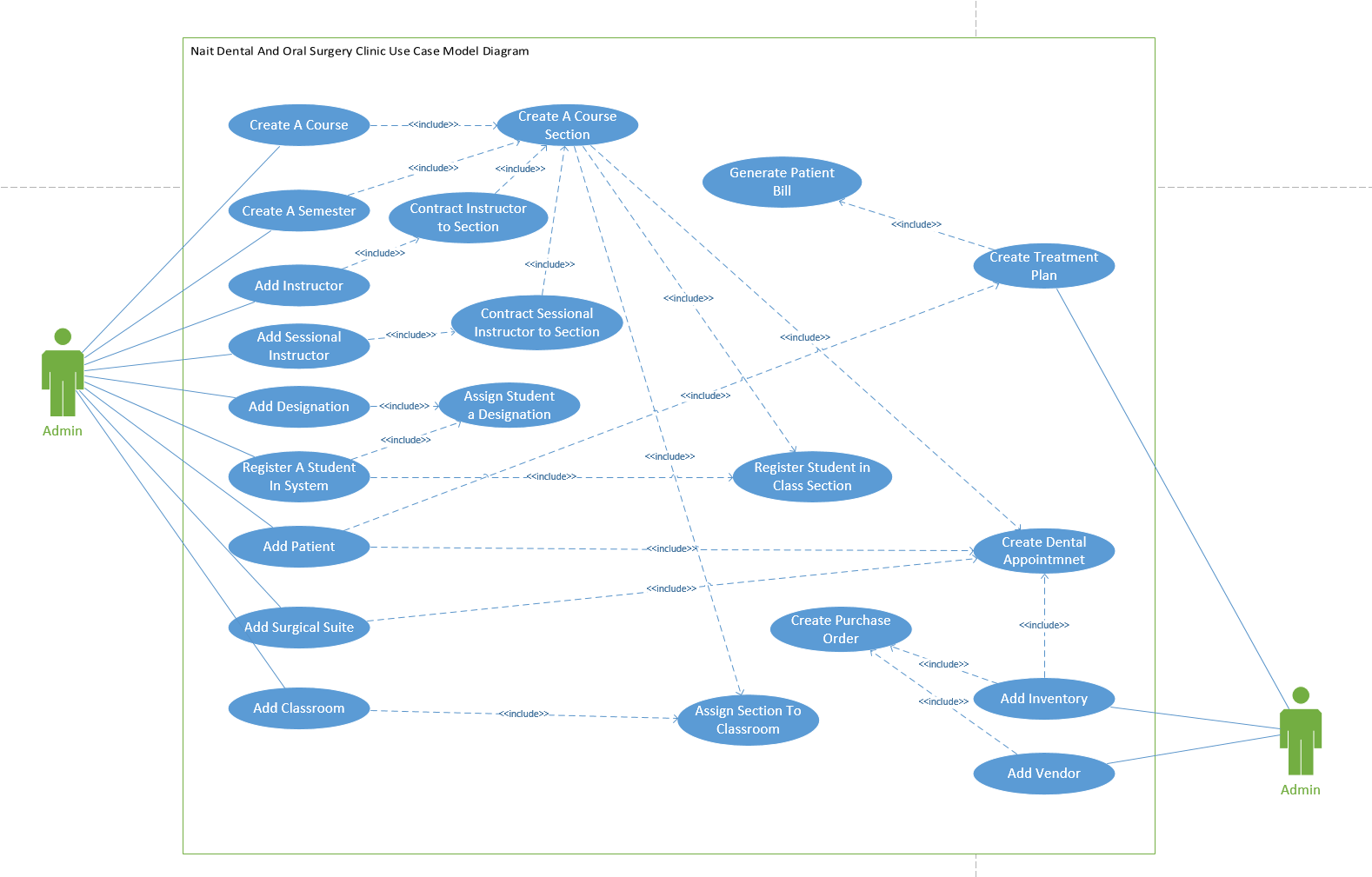
**Benefits to Stakeholders**

* Increased revenue
* Prestige boost
* Increase employment**Risk Analysis**

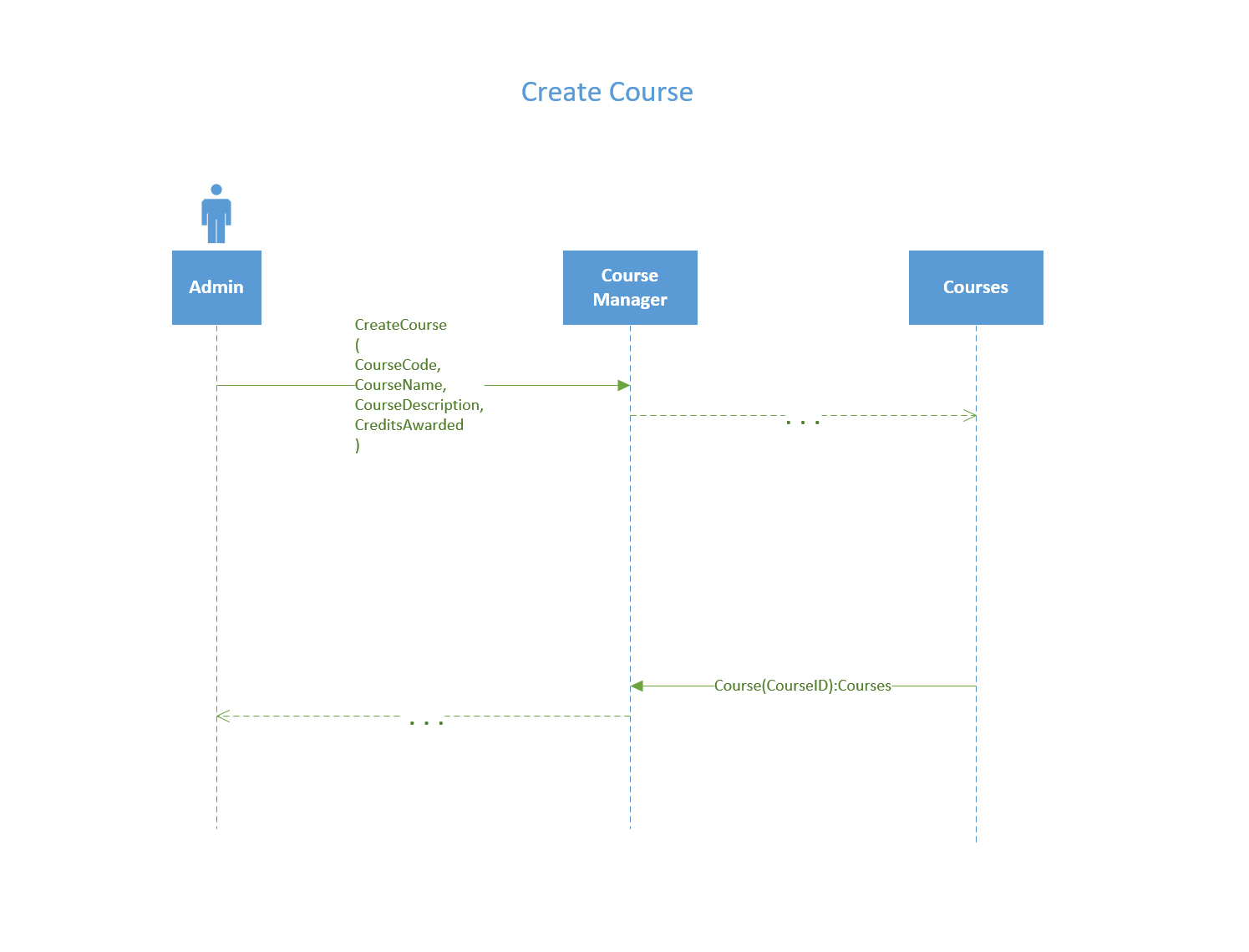
|  |  |  |  |
| --- | --- | --- | --- |
| **Risks** | **Avoidability** | **Impact to Project** | **Reduce Impact** |
| Decrease Revenue |  |  |  |
| Employment |  | Decrease employment that can affect the scheduled sections |  |
| Unsatisfied customer |  | Customers will not be satisfied with the service that will make them choose another clinic and the system will not reach the target income |  |
| Price Competition |  |  |  |
| Decrease in market demand |  |  |  |
| Difficulty for reaching the target due date |  |  |  |

**Summary and Recommendations**

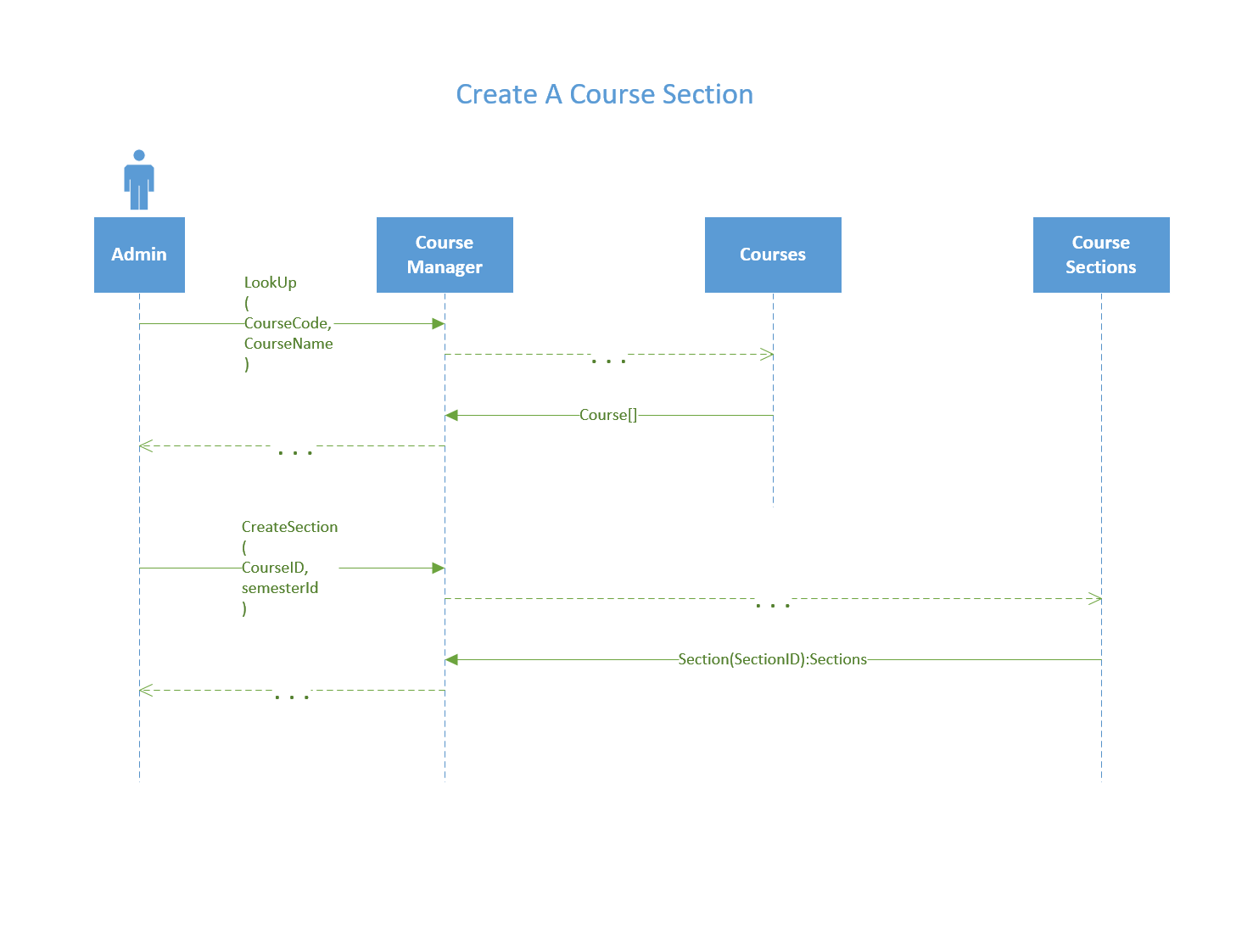
**Diagrams**



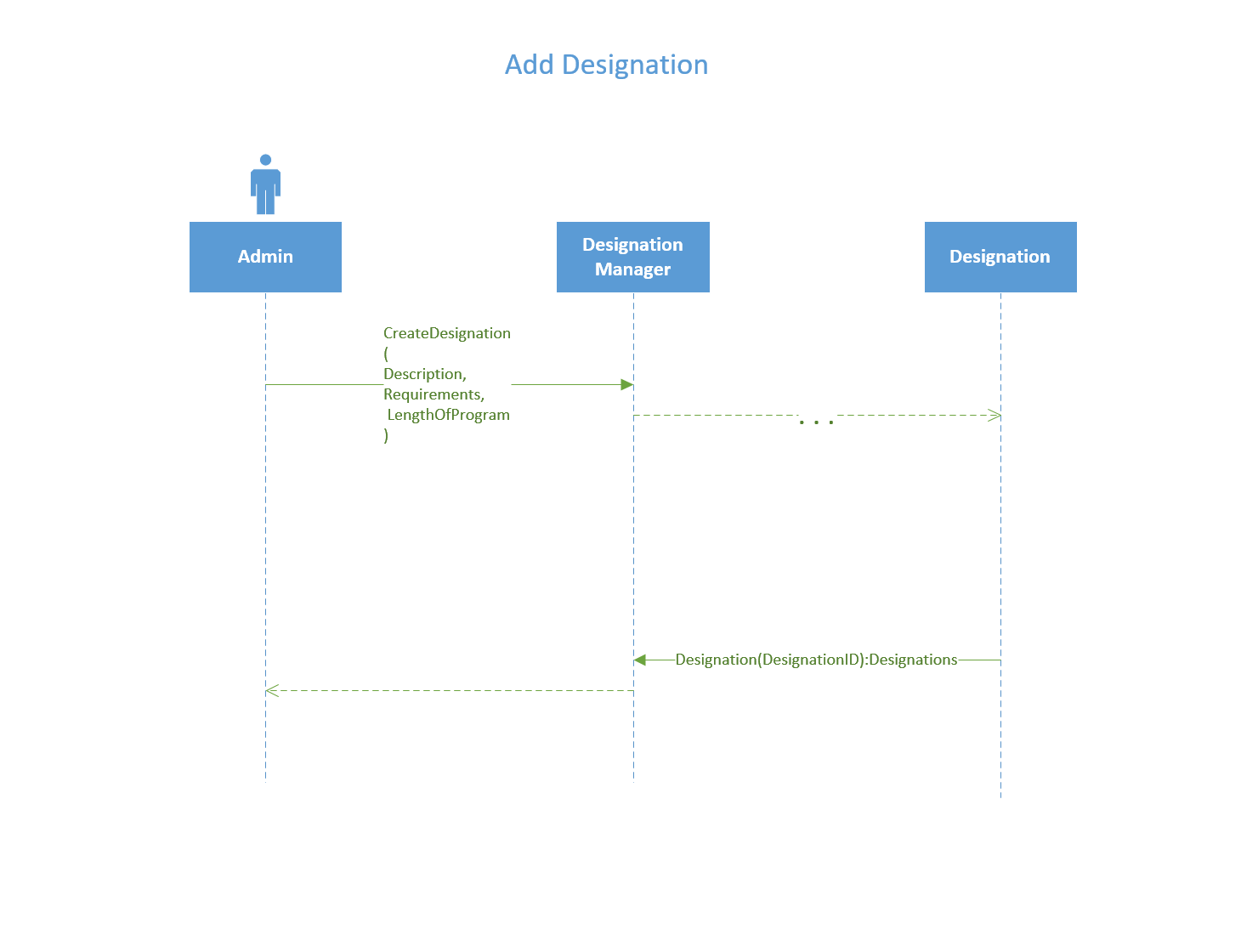
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create Course | |
| **Description:** | Create a course in the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1) CreateCourse(              CourseCode,              CourseName,              CourseDescription,              CreditsAwarded            ) | 2) Course(CourseID) |
| **Post-conditions:** | Course created returns Course object | |
| **Outstanding Issues:** |  | |



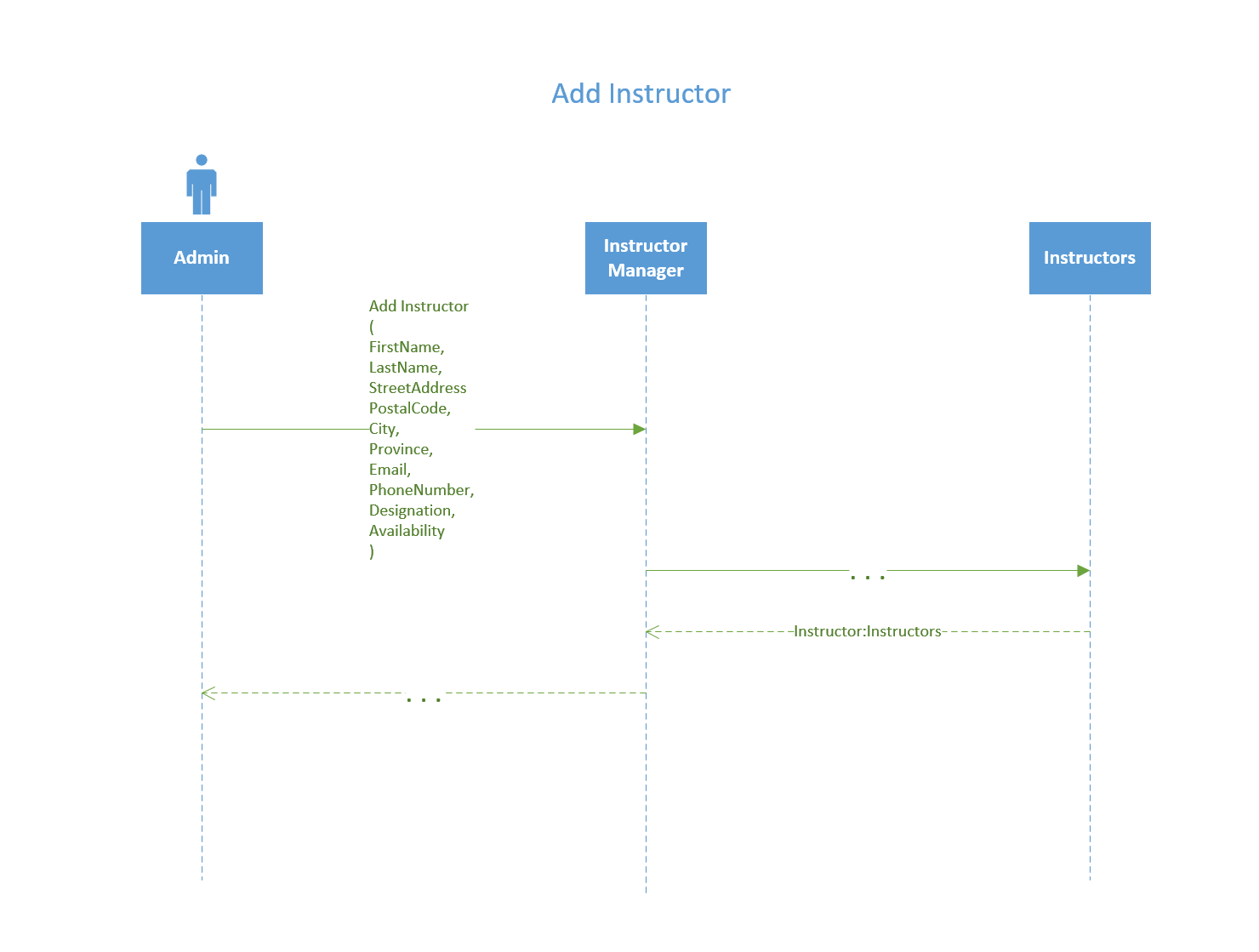
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create A Course Section | |
| **Description:** | Creates a course section in a course in the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Course has been created | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp (               CourseCode,              CourseName           )       3) Create Section(            CourseID,            semesterID            ) | 2) Course[]  4) CourseSection(CourseSectionID) |
| **Post-conditions:** | Section is created | |
| **Outstanding Issues:** |  | |



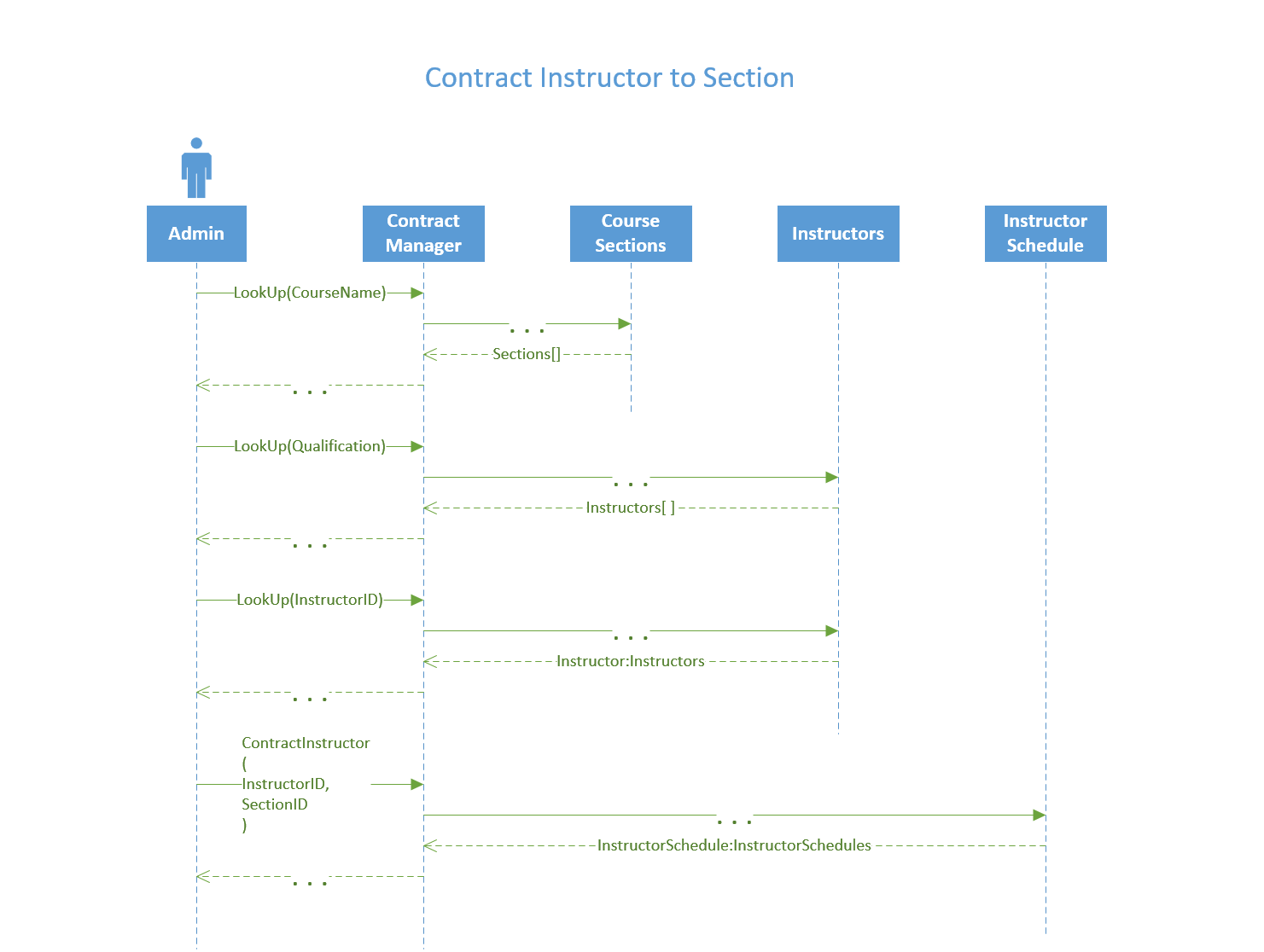
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Designation | |
| **Description:** | Adds a designation into system | |
| **Actor(s):** | admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. CreateDesignation(               Description,              requirements,              LengthOfProgram            ) | 2) Designation(DesignationID) |
| **Post-conditions:** | * Designation added | |
| **Outstanding Issues:** |  | |



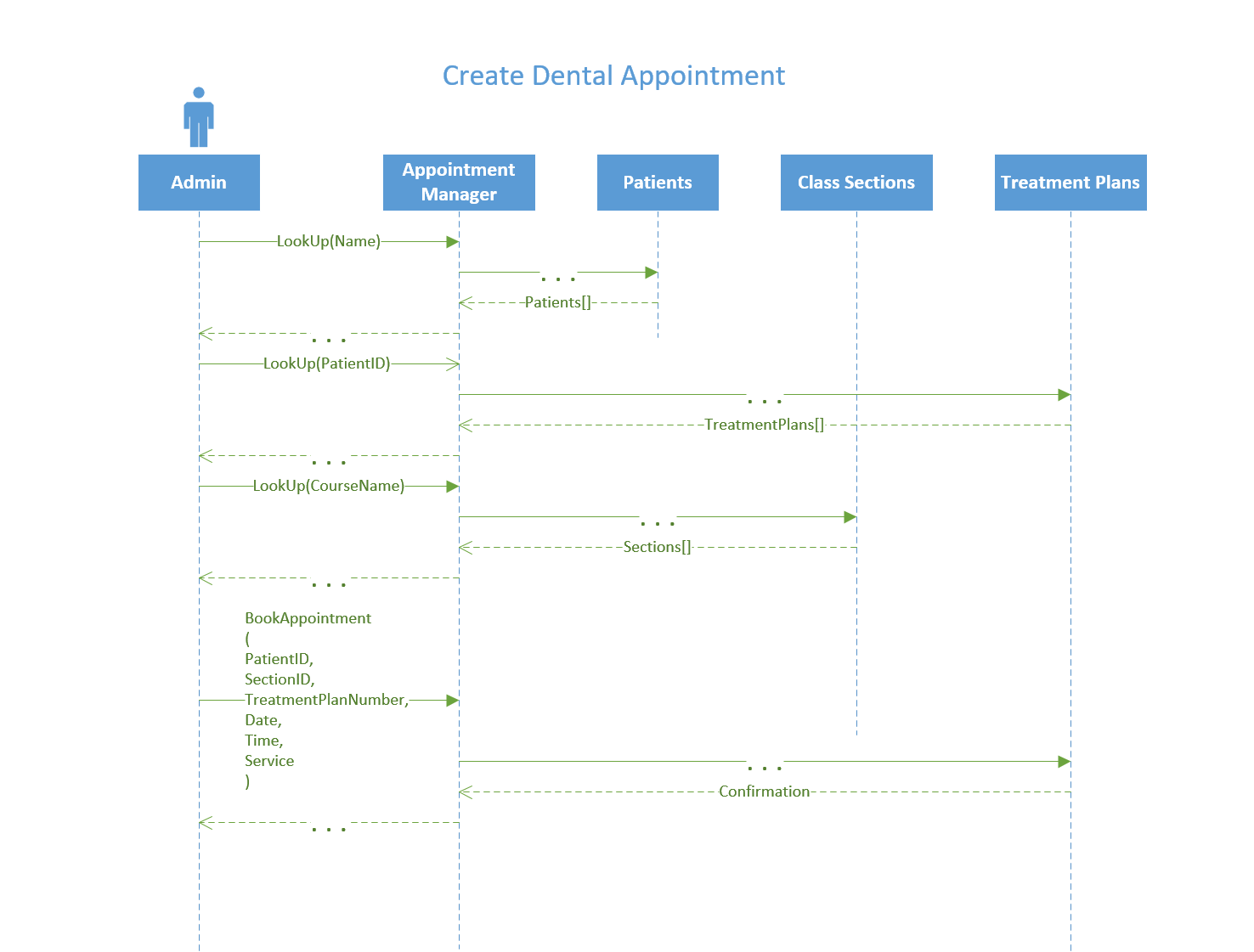
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Instructor | |
| **Description:** | Adds a instructor to system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. Add Instructor(               FirstName,              LastName,              StreetAddress,              PostalCode,              City,              Province,              Email,              PhoneNumber,              Designation,              Availability            ) | 2) Instructor(InstructorID) |
| **Post-conditions:** | * Instructor added | |
| **Outstanding Issues:** |  | |



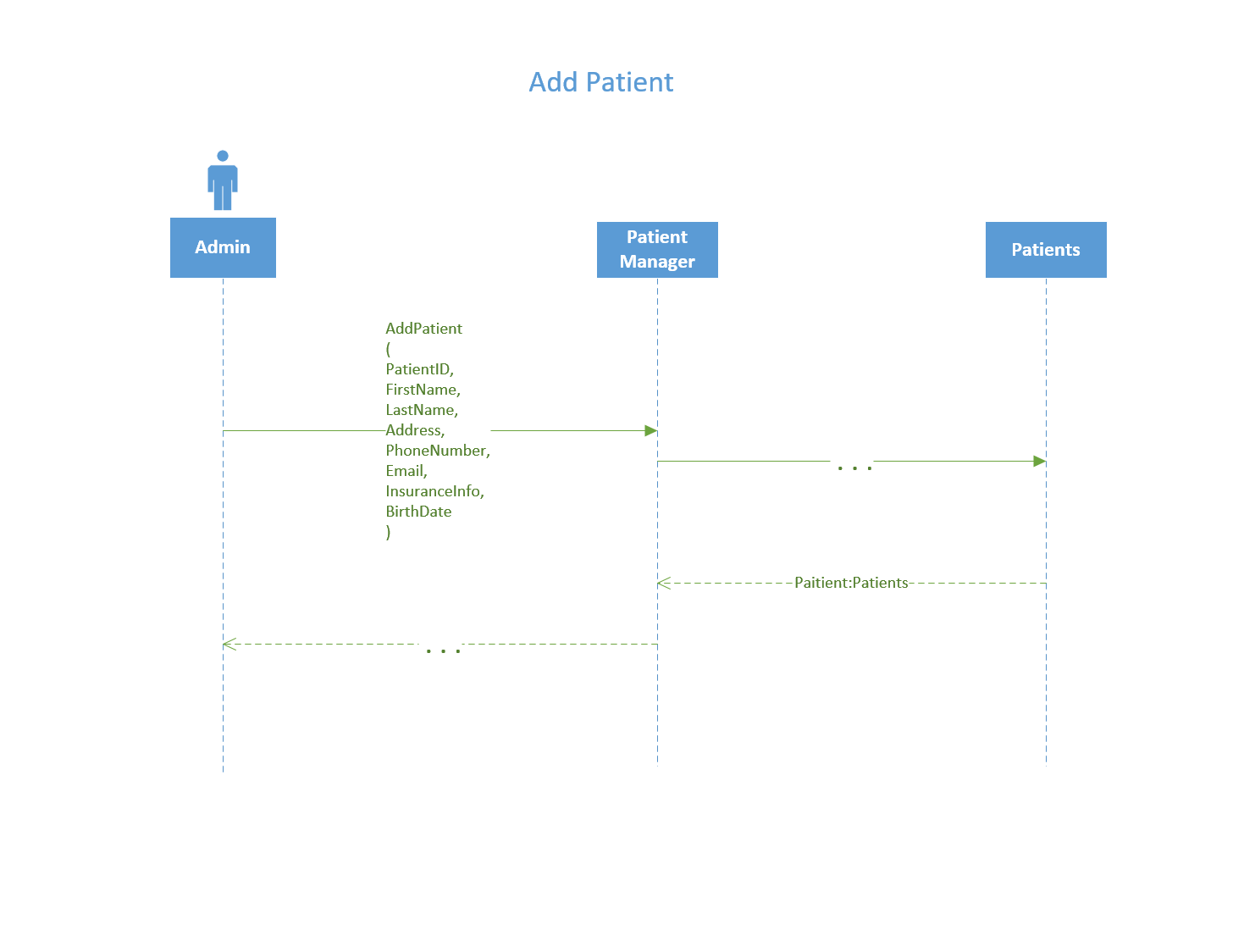
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Contract Instructor to Section | |
| **Description:** | Look up list courses and list of qualified instructors before assigning an instructor to a course section. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | * Section is created * Instructor exist | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(CourseName)           3)  LookUp(Designation)      5) LookUp(InstructorID)      7) ContractInstructor(              InstructorID,              SectionID          ) | 2) Sections[]  4) Instructors[]  6)Instructor:Instructors  8)Section:Sections |
| **Post-conditions:** | * Instructor assigned to course section | |
| **Outstanding Issues:** |  | |



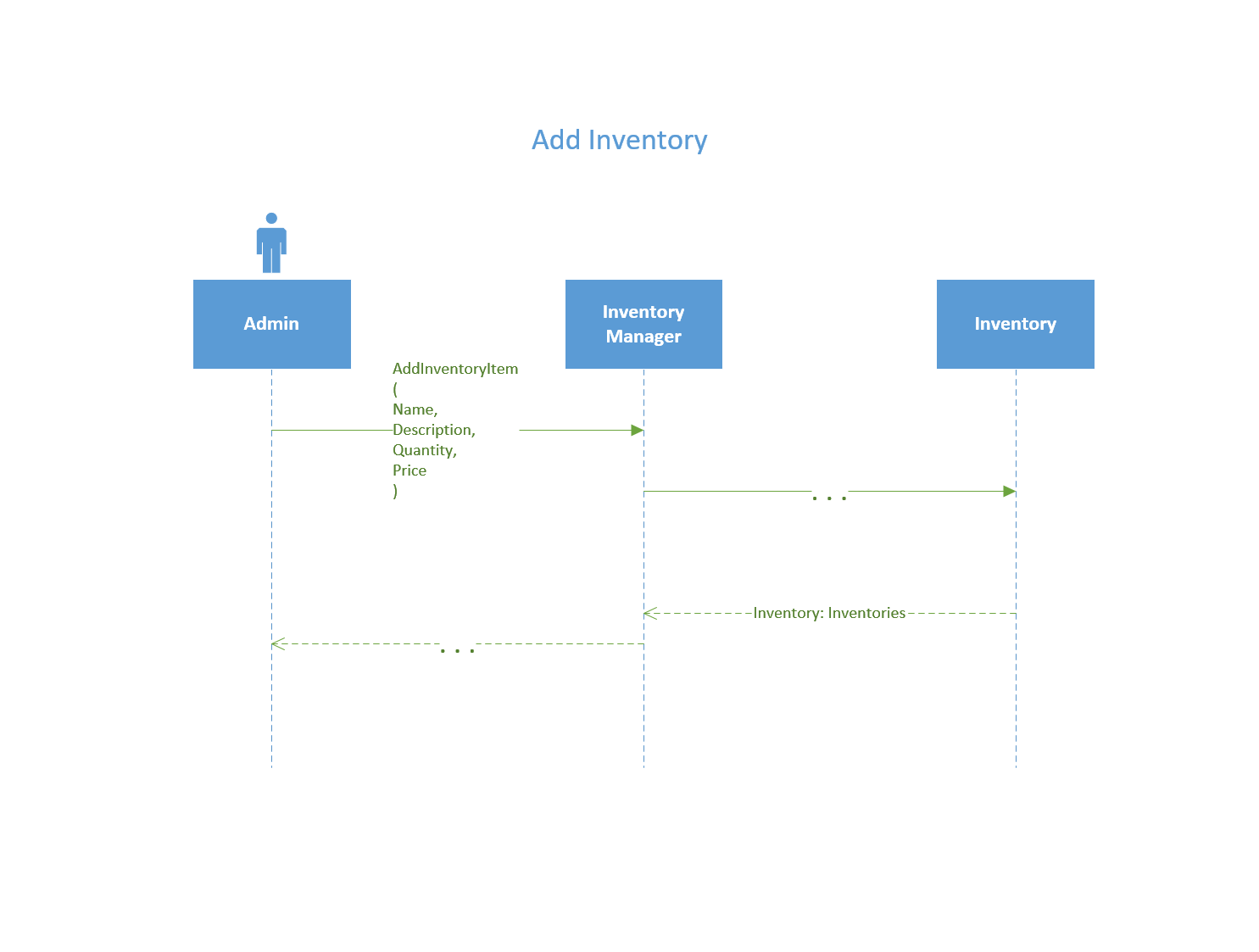
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create Dental Appointment | |
| **Description:** | Looks up a patient and their treatment plans, as well as a list of section to book the appointment with. Then books the appointment. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | * Patient Exist * TreatmentPlan Exists * Student Exist * Available Surgical Suite * Available inventory | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(Name)       3) LookUp(PatientID)      5) LookUp(CourseName)      7) BookDentalAppointment          (            PatientID,            SectionID,            SuiteID,            treatmentPlanNumber,            Date,            Time,            Service  ) | 2) Patients[]  4)TreatmentPlans[]  6)Sections[]  8)Confirm |
| **Post-conditions:** | * Confirm | |
| **Outstanding Issues:** |  | |



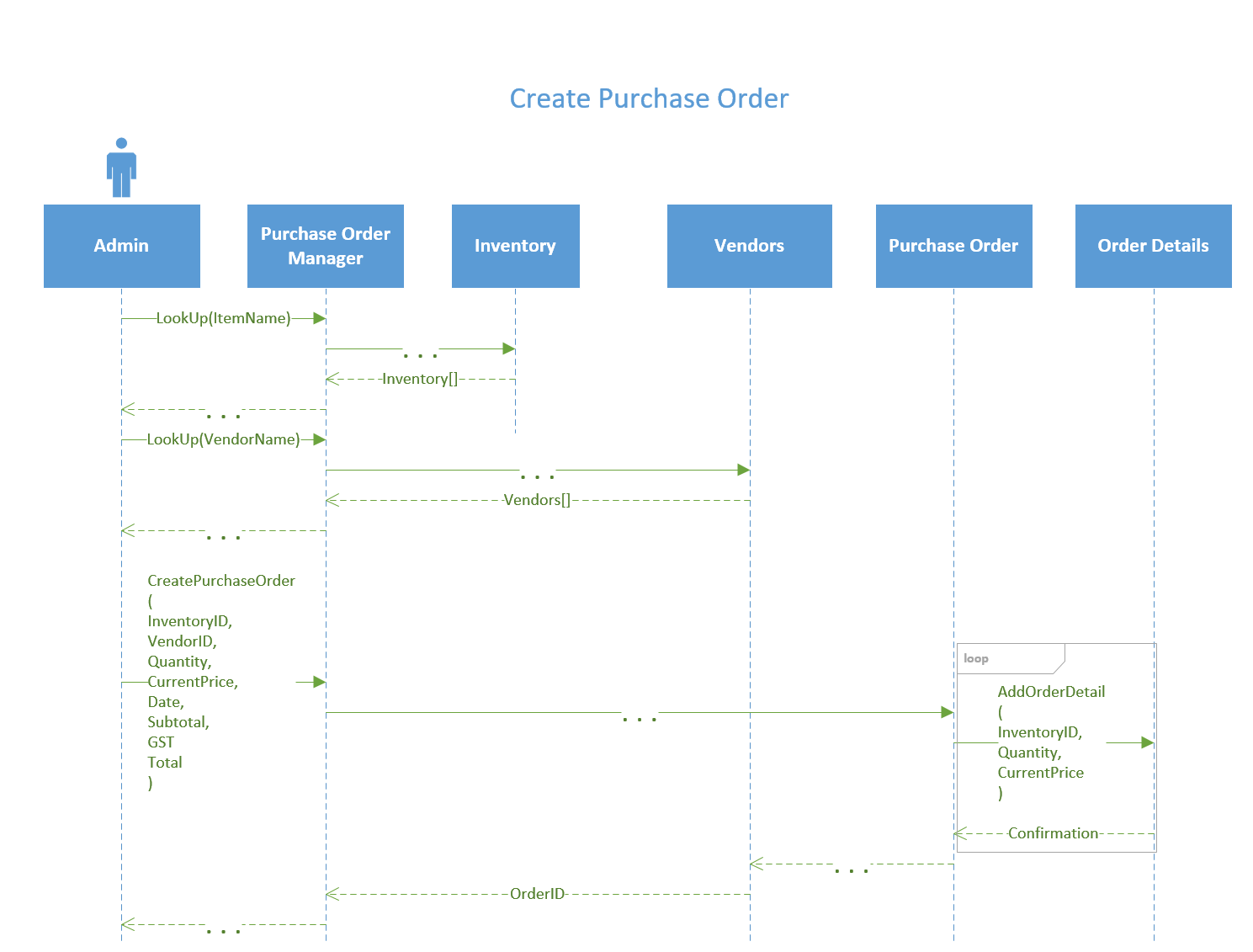
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Patient | |
| **Description:** | Adds a Patient to the system. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Patient is not in system | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddPatient (              PatientID,             FirstName,             LastName,             Address,             PhoneNumber,             Email,             InsuranceInfo,             DateOfBirth           ) | 2) Patient:Patients |
| **Post-conditions:** | Patient is added to the system | |
| **Outstanding Issues:** |  | |



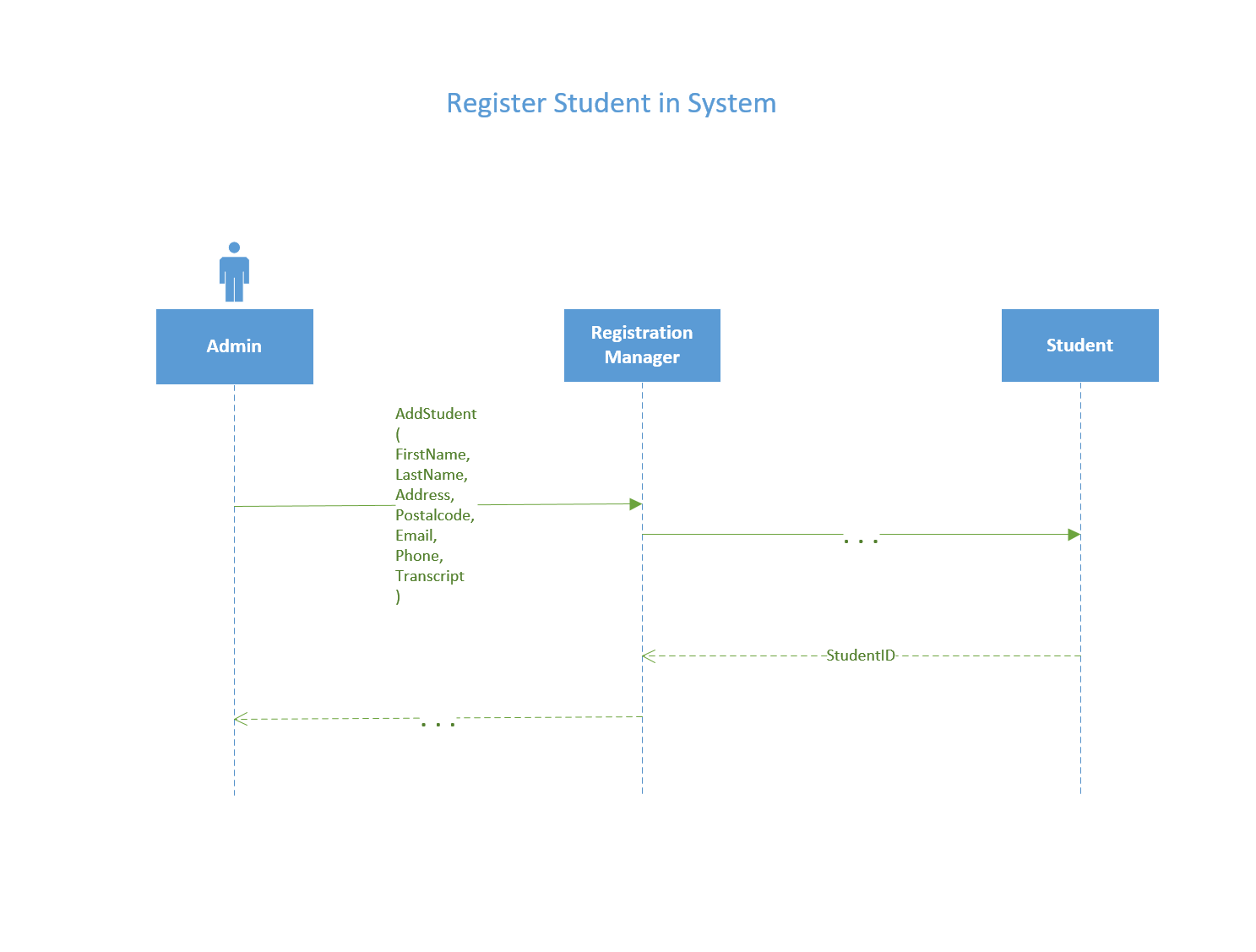
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Inventory | |
| **Description:** | Process is used to add inventory to the system | |
| **Actor(s):** | admin | |
| **Preconditions:** | itemID must not already exist in the system | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddInventoryItem (                Name,               Description,               Quantity,               Price           ) | 2)  Inventory : Inventories |
| **Post-conditions:** | A new item will be added to the inventory table and assigned a unique ID number. | |
| **Outstanding Issues:** |  | |



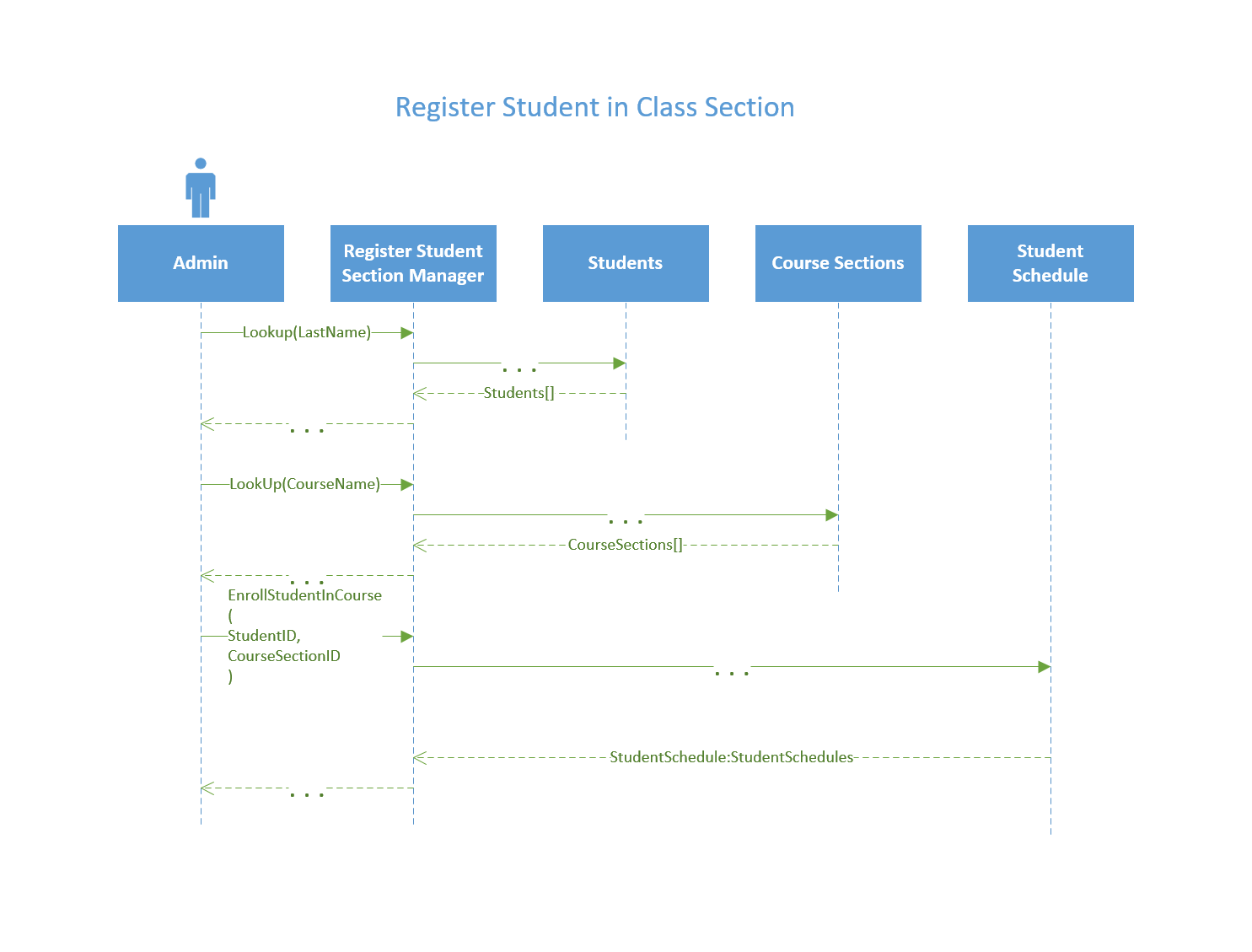
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create a Purchase Order | |
| **Description:** | Looks up and Items and vendors so the admin can create an order. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Item needs to exist in the system, vendor must be in the system | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(ItemName)         3) LookUp(VendorName)        5)  createPurchaseOrder(              InventoryID,              VendorID,              Quantity,              CurrentPrice,              Date,              Subtotal,              GST,              Total           )      7) AddOrderDetails(             InventoryId,             Quantity,             CurrentPrice          ) | 2) Inventory[]  4) Vendors[]  6) purchaseOrder(purchaseOrderID) |
| **Post-conditions:** | A complete purchase order is created with a unique ID | |
| **Outstanding Issues:** |  | |



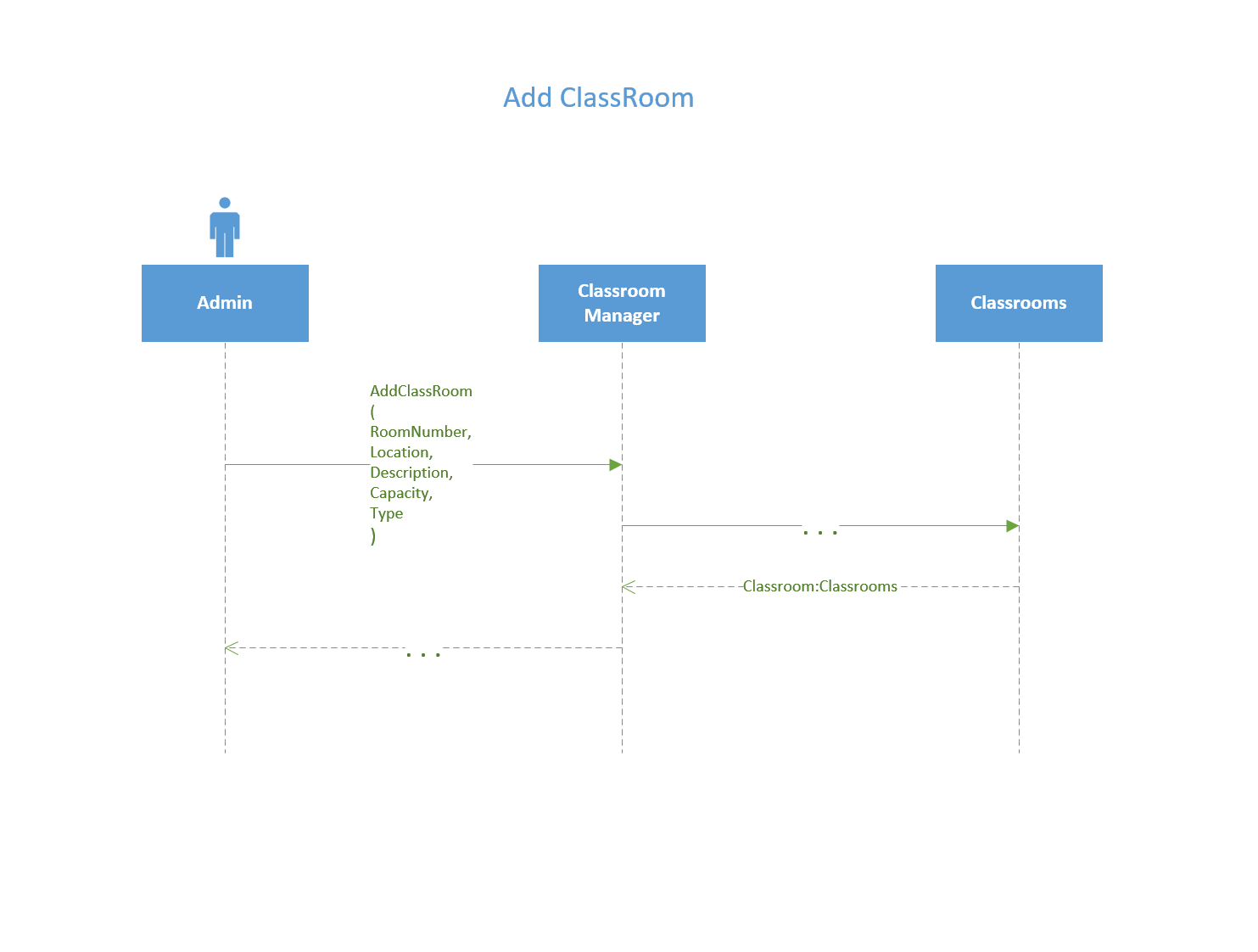
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Register Student in System | |
| **Description:** | Registers a Student in the system. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Student does not exist in system | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddStudent(             FirstName,            LastName,            Address,            Postal,            Email,            Phone,            Transcript           ) | 2) Student:Students |
| **Post-conditions:** | A new student is added to the system under a unique Student ID | |
| **Outstanding Issues:** |  | |



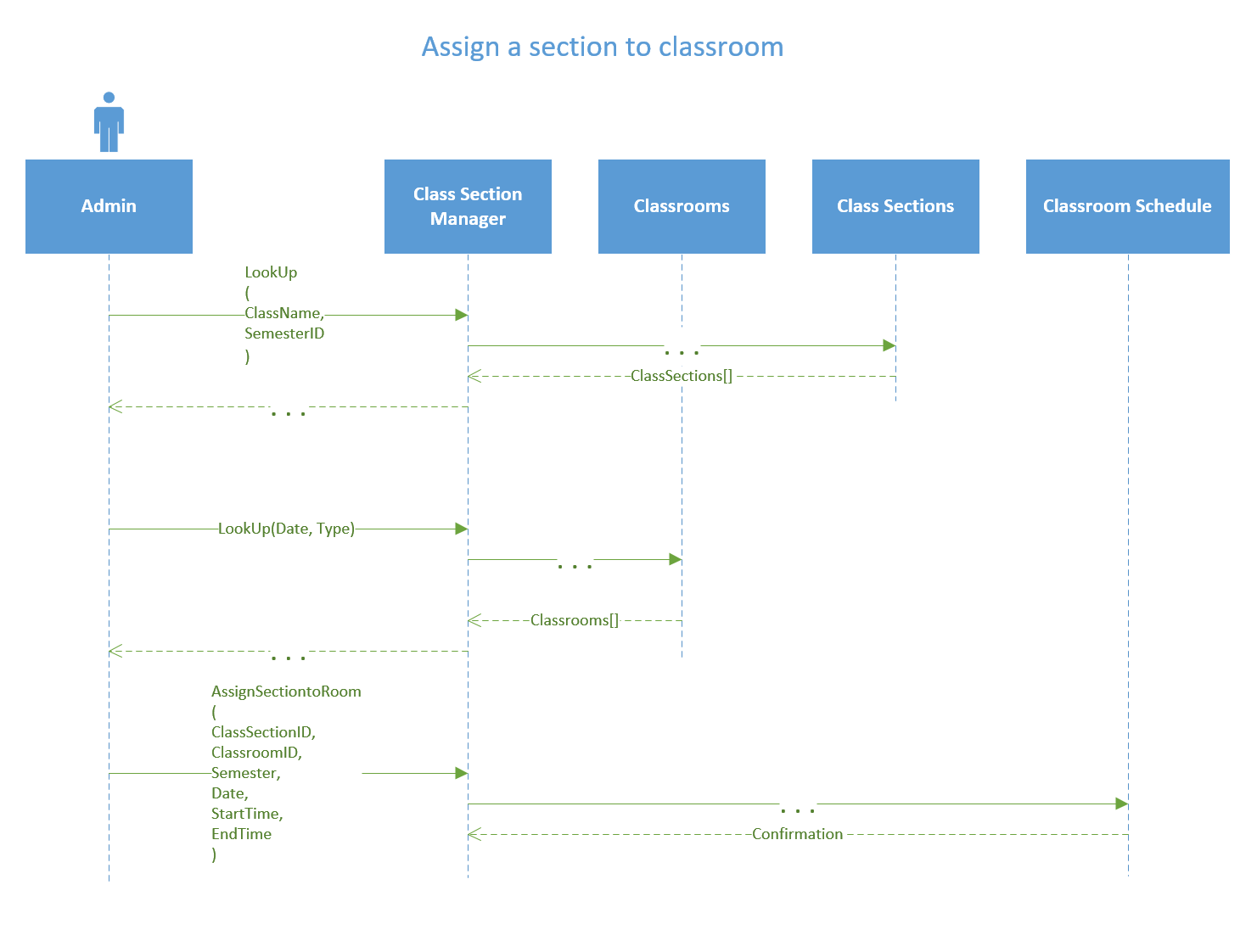
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Register Student in Class Section | |
| **Description:** | Registers a student into a class section | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Section must exist, Student Exists, space must be available, no time block overlap in student’s schedule | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(LastName)       3)  LookUp(CourseName)      5)  EnrollStudentInCourse(             StudentID,             CourseSectionID           ) | 2) Students[]  4) CourseSections[]  6) UpdateClassList(       StudentID,       CourseSectionID     )  7) Classlist:Classlists |
| **Post-conditions:** | Student is enrolled in class section | |
| **Outstanding Issues:** |  | |



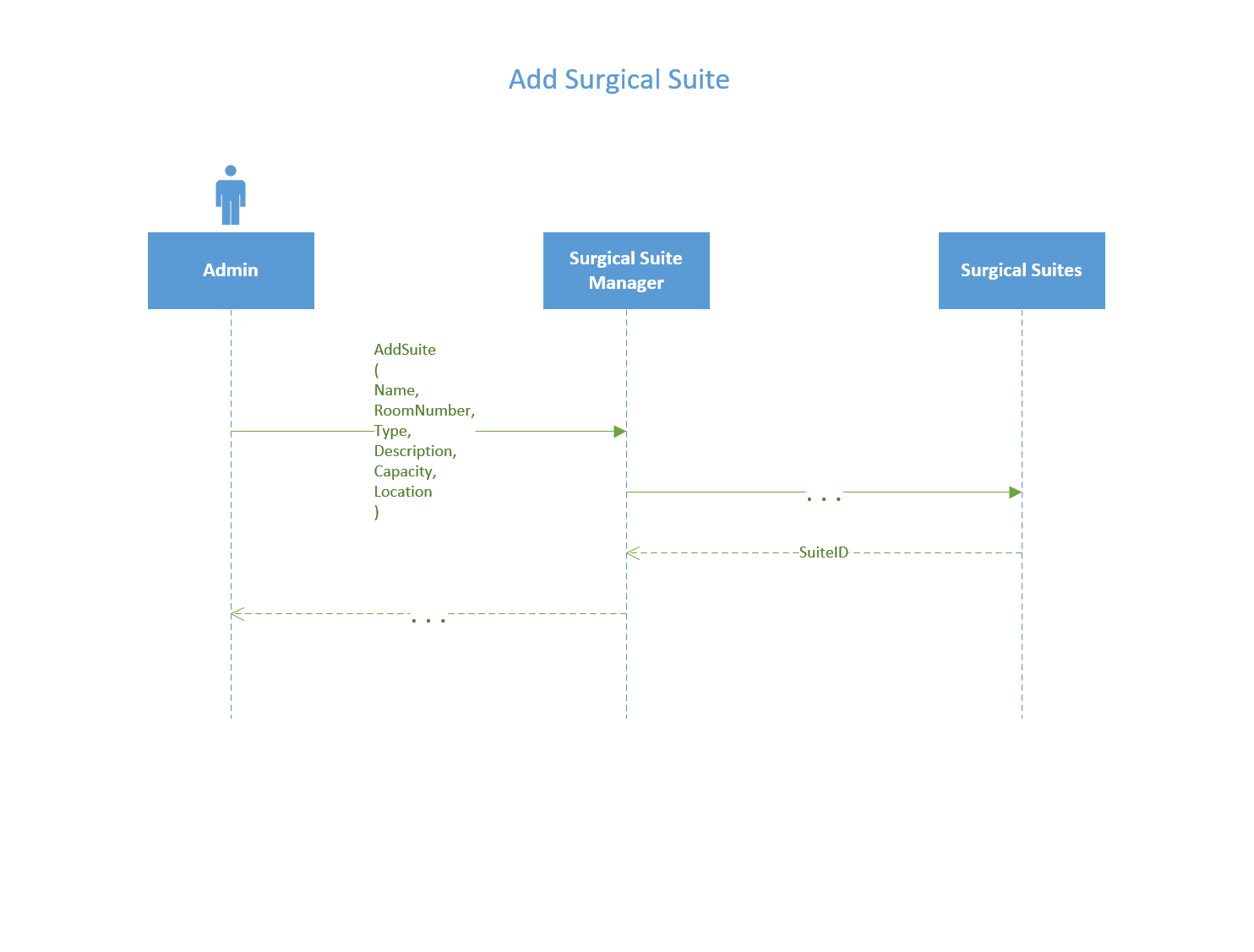
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Class Room | |
| **Description:** | This process would be used to add a single instance of a classroom to the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Location does not exist | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddClassRoom (             RoomNumber,            Location,            Description,            Capacity,            Type           ) | 2) Classroom:Classrooms |
| **Post-conditions:** | A classroom is added to the system and giving a unique ID number | |
| **Outstanding Issues:** |  | |



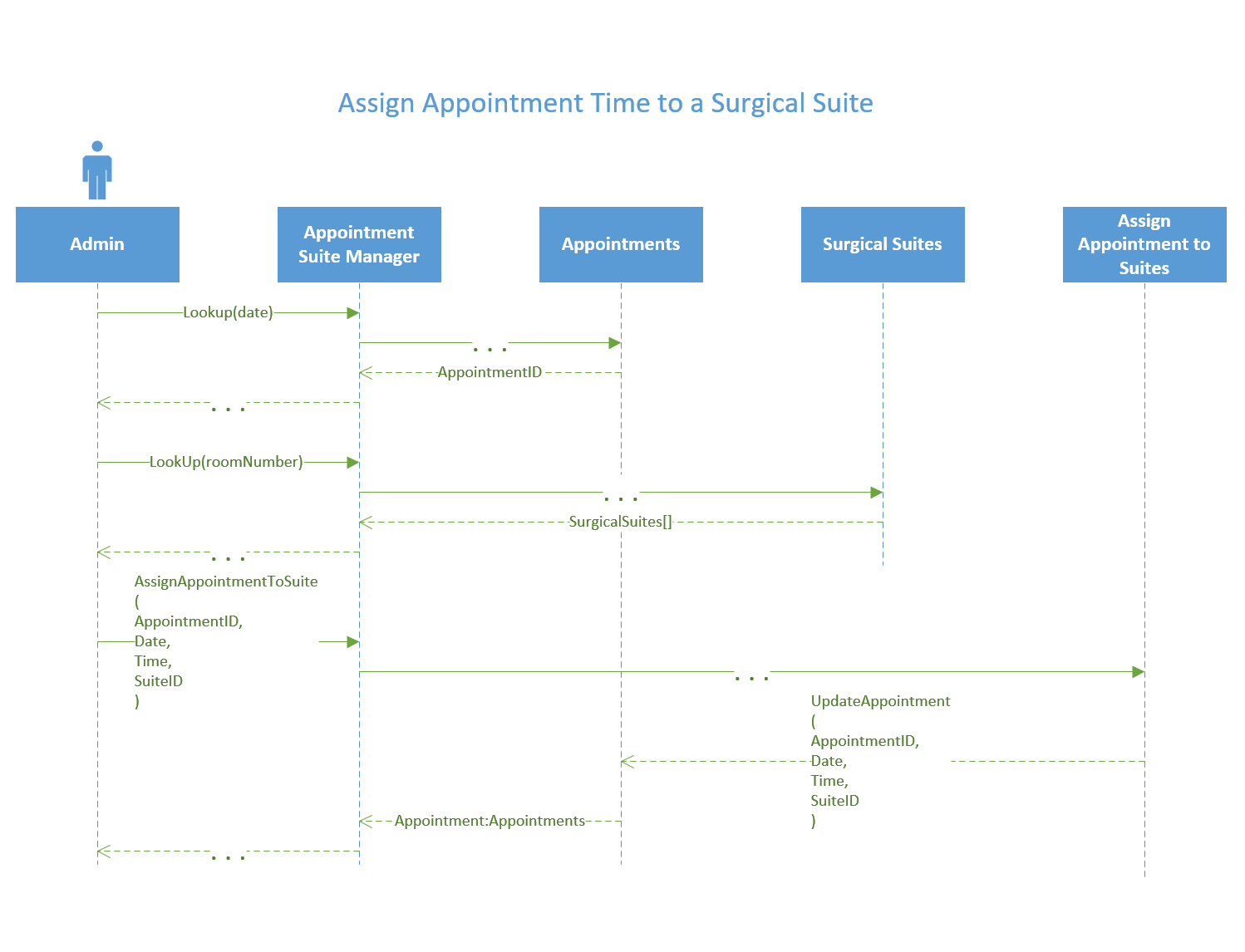
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Assign a section to classroom | |
| **Description:** | Admin looks up sections and classrooms and assigns section to an available room. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Room is added, Room schedule is created, Section has no room, Section is created | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(               ClassName,              Semester,           )      3) LookUp(Date, Type)      5) AssignSectionToRoom(               ClassroomID,               SectionID,               Semester,               Date,               StartTime,               EndTime               ) | 2) ClassSections[]  4) Classrooms[]  6) Confirmation |
| **Post-conditions:** | Section is added to classroom | |
| **Outstanding Issues:** |  | |



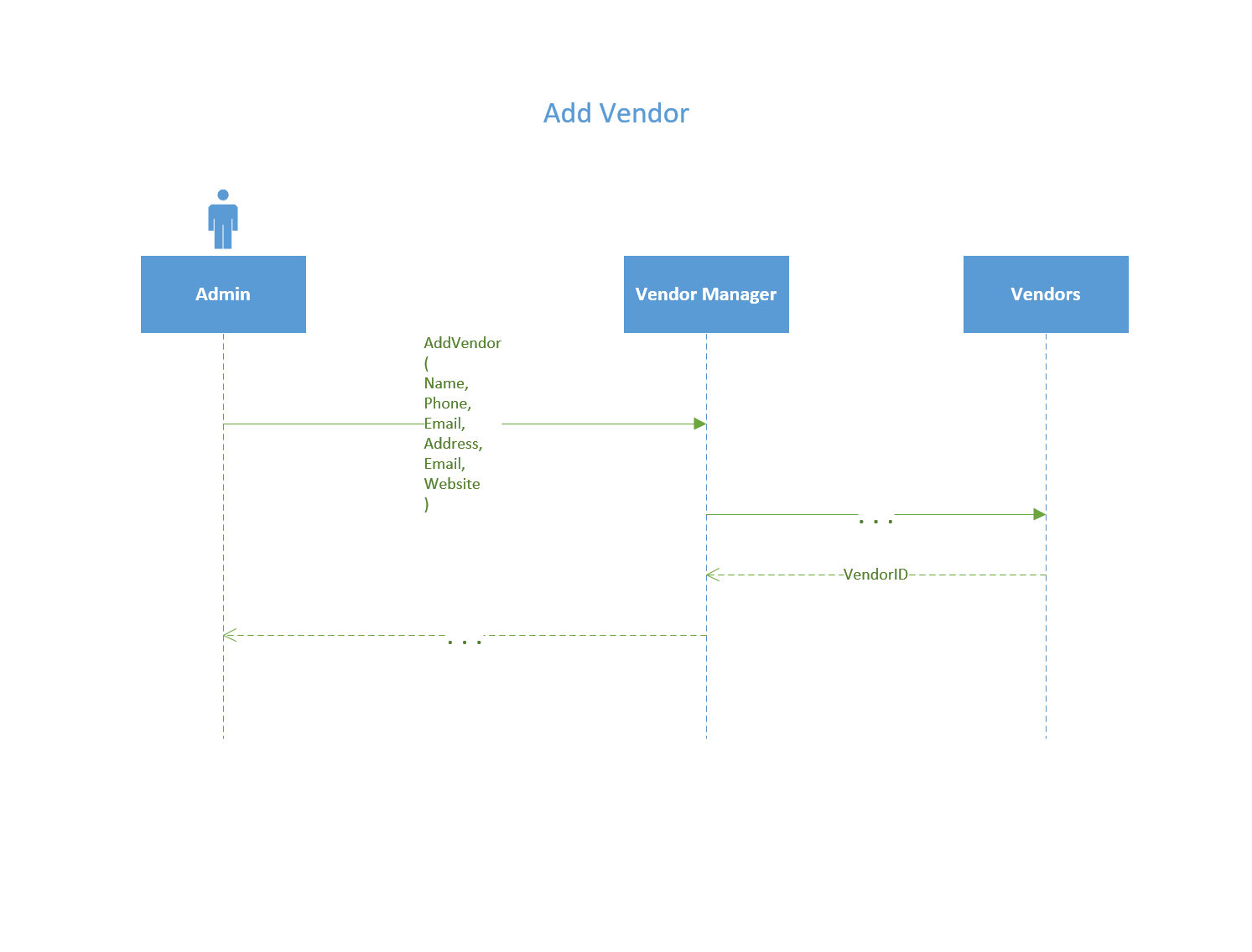
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add surgical Suite | |
| **Description:** | Adds a surgical suite to the system. | |
| **Actor(s):** | admin | |
| **Preconditions:** | Location does not exist | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddSuite(              Name             RoomNumber,             Type,             Tescription,             Capacity,             Location            ) | 2) SuiteID |
| **Post-conditions:** | Returns Suite ID | |
| **Outstanding Issues:** |  | |



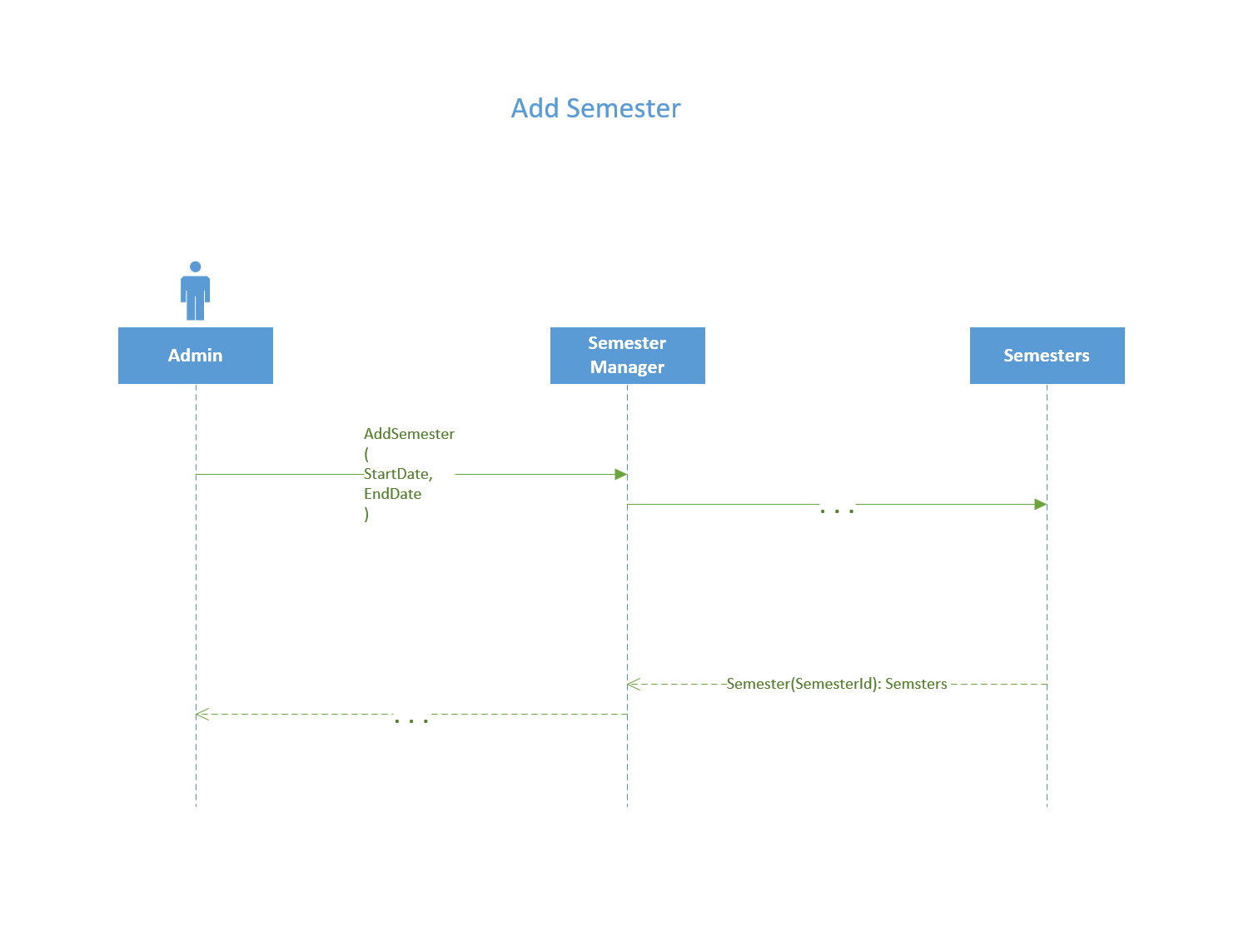
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Assign Appointment Time To Surgical Suite | |
| **Description:** | Updates an appointment with a surgical suite | |
| **Actor(s):** | Admin | |
| **Preconditions:** | * Appointment ID is needed (appointmnetID), a surgical suite must be in the system (suiteID) | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(Date)       3) LookUp(RoomNumber)    5) AssignAppointmentToSuite        )           AppointmentID,           Date,           Time,           SuiteID        ) | 2) AppointmentID  4) SurgicalSuites[]  6) UpdateAppointment     (       AppointmentID,       Date,       Time,       SuiteID     )  7) Appointment:Appointments |
| **Post-conditions:** | * Suite is booked on the time and date for the appointmentID | |
| **Outstanding Issues:** |  | |



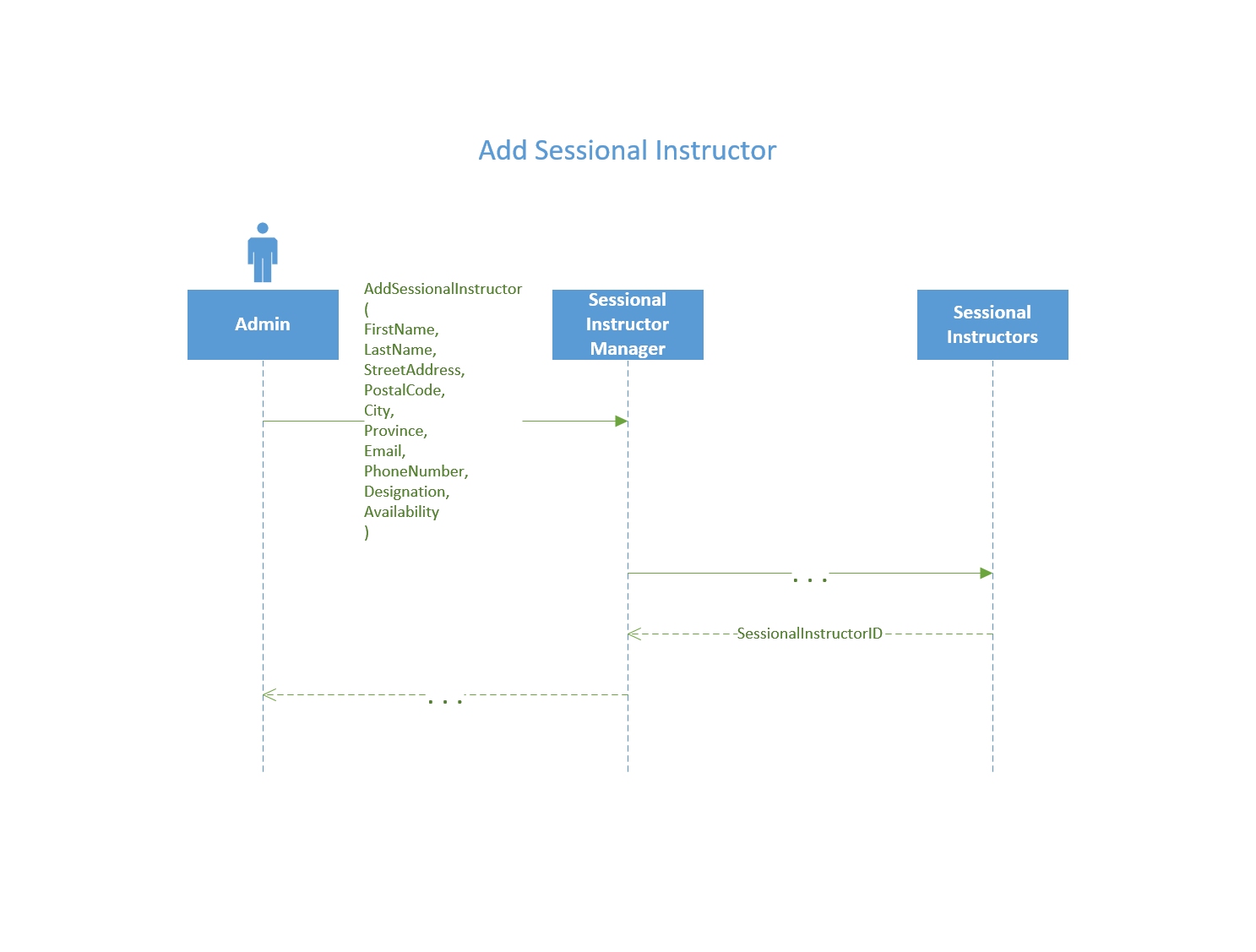
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Vendor | |
| **Description:** | Adds a new vendor to the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddVendor           (              Name,              Phone,              Email,              Address,              Email,              Website           ) | 2) VendorID |
| **Post-conditions:** | A new vendor is added into the system and giving an ID | |
| **Outstanding Issues:** |  | |



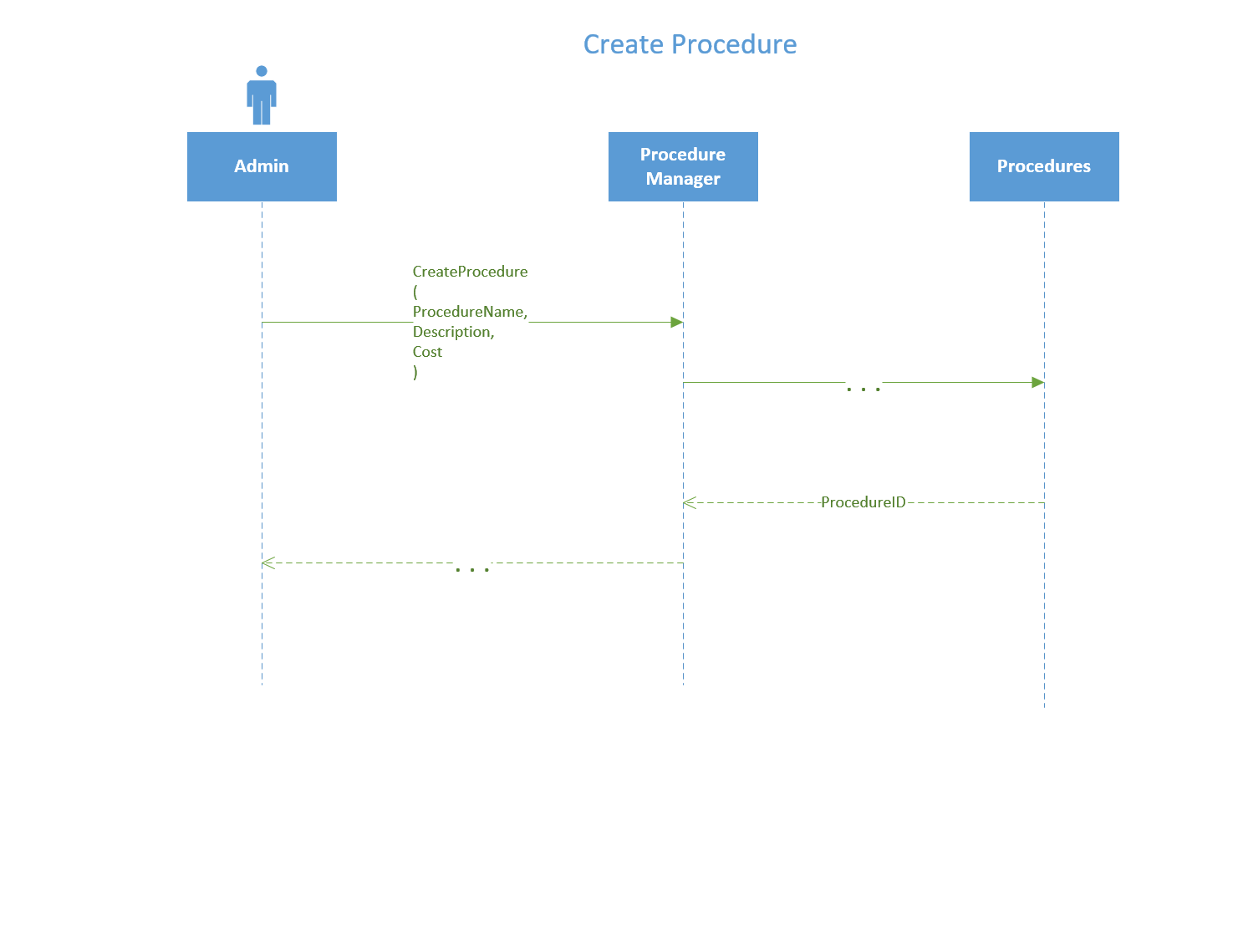
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Semester | |
| **Description:** | Adds a new semester to the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. AddSemester           (    StartDate,    EndDate  ) | 2) Semester:Semesters |
| **Post-conditions:** | Semester added to system and Id created. | |
| **Outstanding Issues:** |  | |



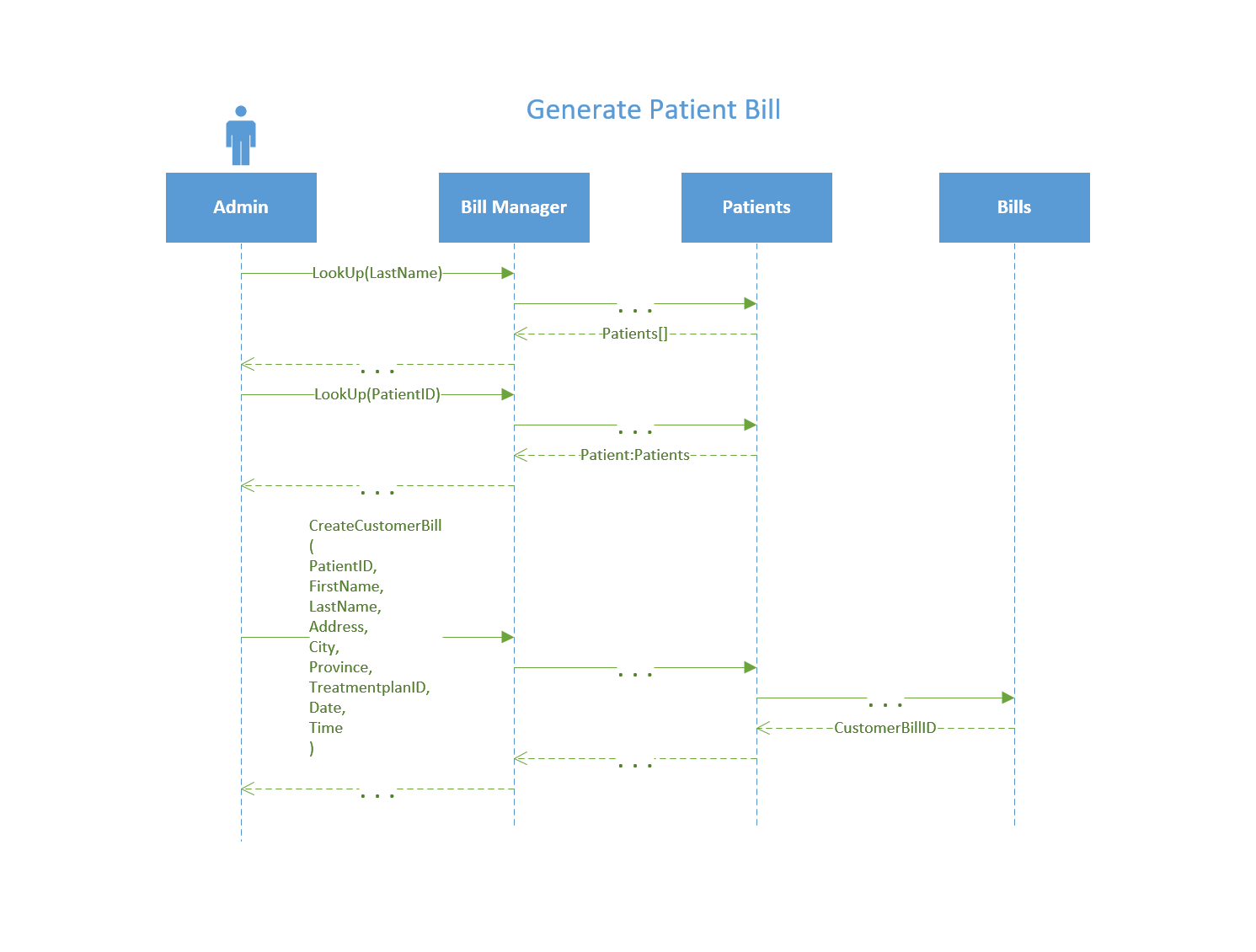
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Add Sessional Instructor | |
| **Description:** | Adds a new sessional instructor to the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1)AddSessionalInstructor     (  FirstName,  LastName,  StreetAddress,  PostalCode,  City,  Province,  Email,  PhoneNumber,  Designation,  Availability     ) | 2) SessionalInstructorID |
| **Post-conditions:** | SessionalInstructorID | |
| **Outstanding Issues:** |  | |



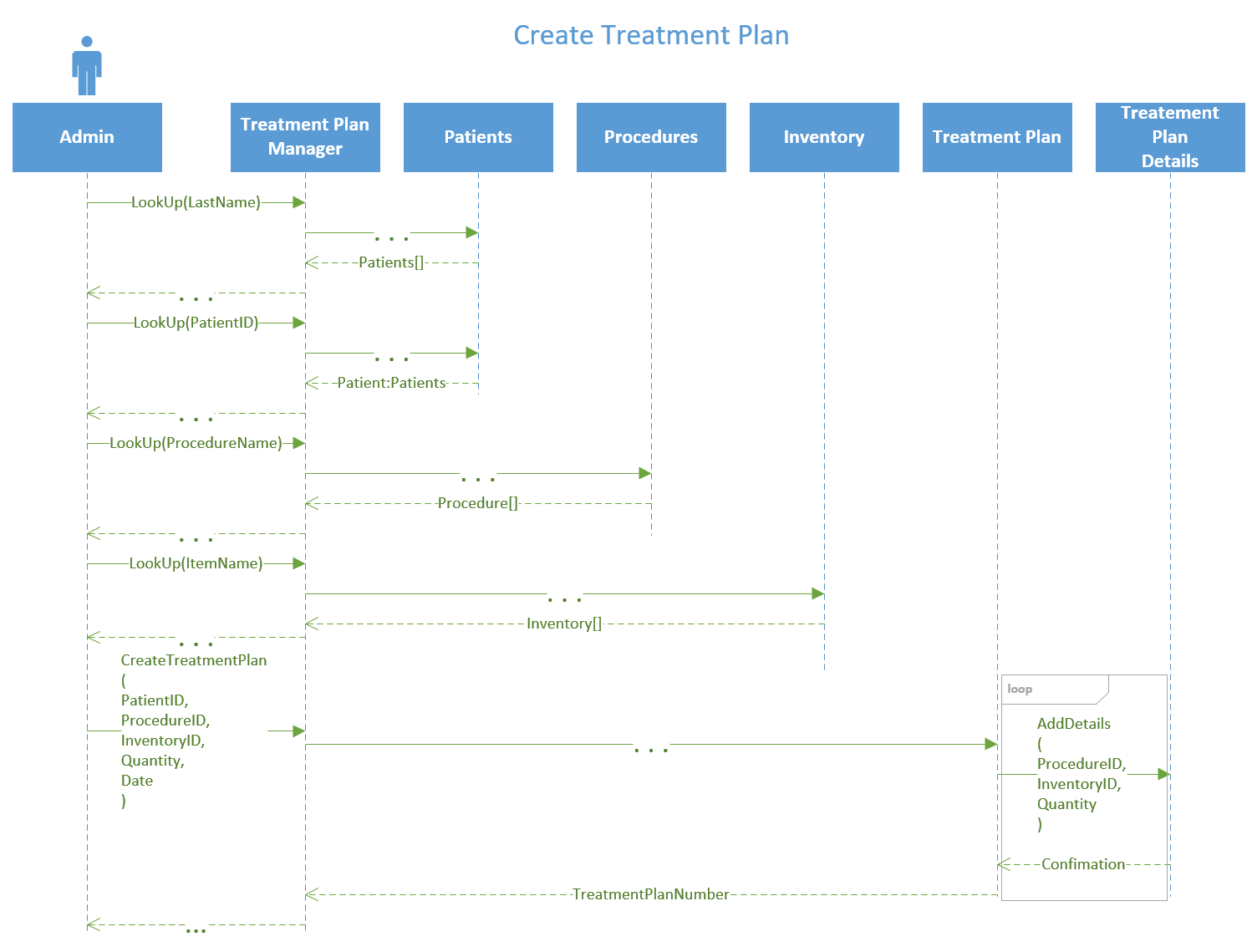
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create Procedure | |
| **Description:** | Adds a new procedure to the system | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. CreateProcedure   (  ProcedureName,  Description,  Cost  ) | 2) ProcedureID |
| **Post-conditions:** | ProcedureID | |
| **Outstanding Issues:** |  | |



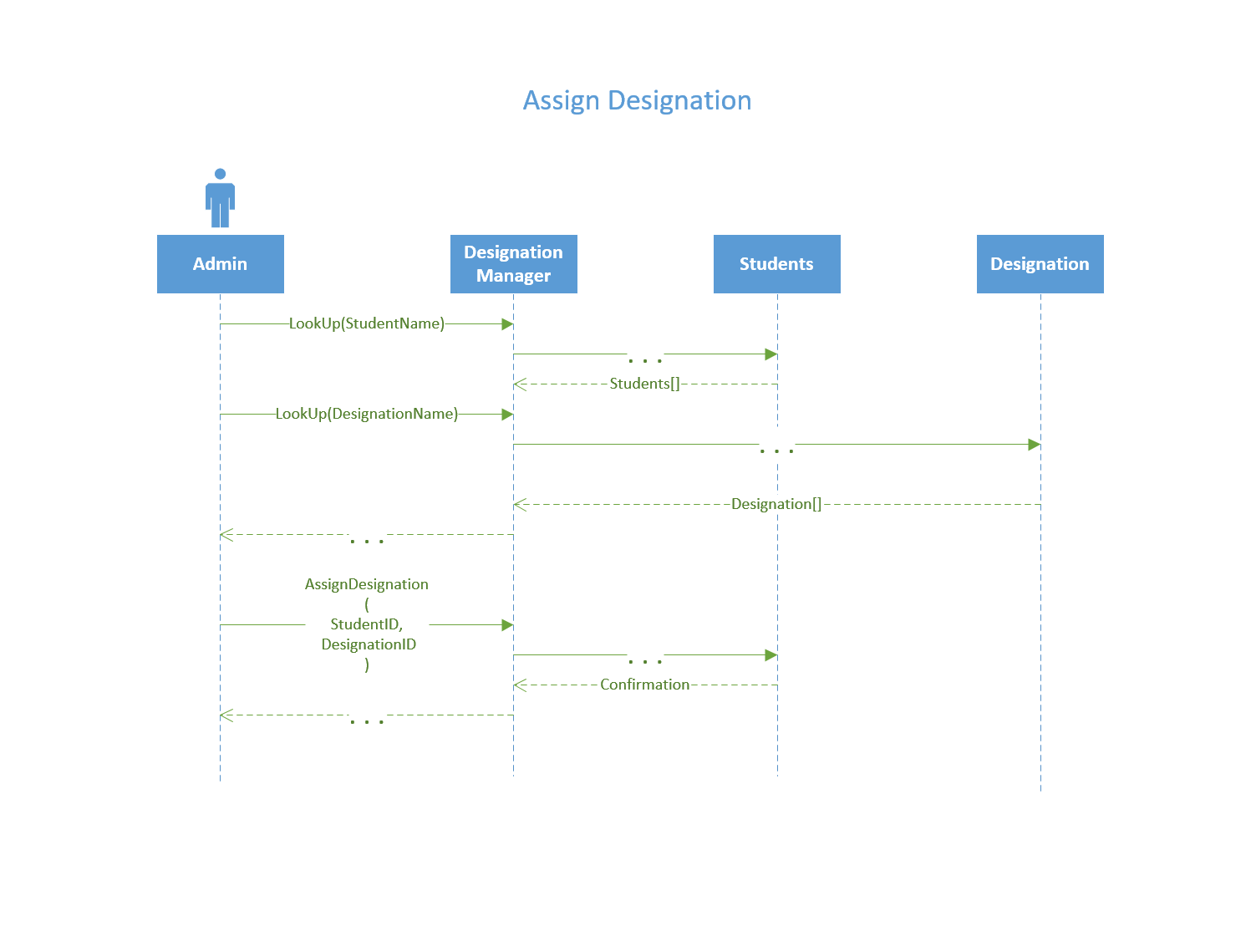
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Generate Patient Bill | |
| **Description:** | Admin looks up patient information a enter information into system to generate customer bill. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | -Patient Exists  -At least one Appointment | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1) LookUp(lastName)  3) LookUp(PatientID)  5)CreateCustomerBill    (        PatientID,        FirstName,        LastName,        Address,        City,        Province,        TreatmentplanID,        Date,        Time     ) | 2) Patients[]  4) Patient:Patients  6) CustomerBillID |
| **Post-conditions:** | Bill returned | |
| **Outstanding Issues:** |  | |



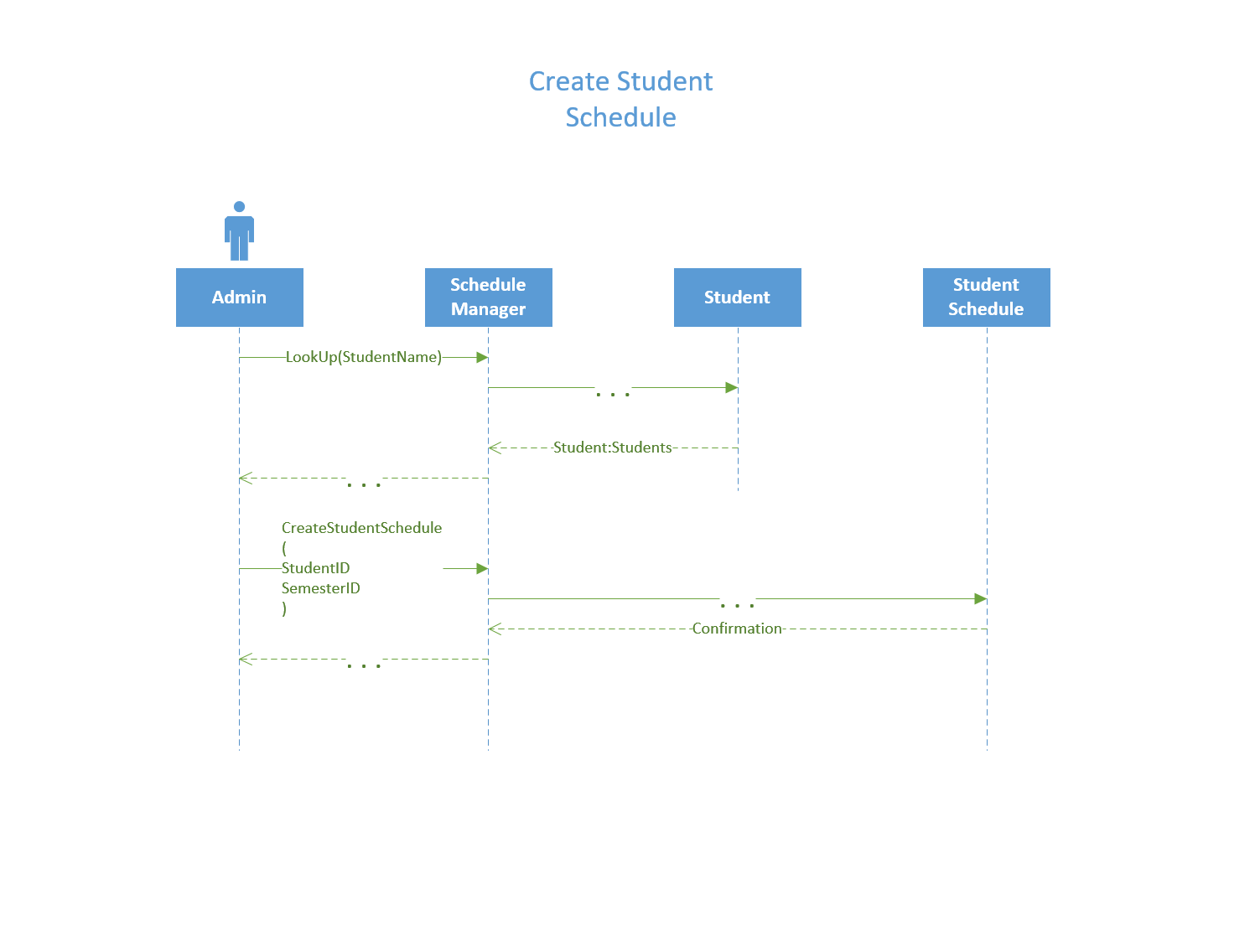
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create Treatment Plan | |
| **Description:** | Admin looks up the patient and inventory to create treatment plan. Creating a treatment plan adds details to procedures which are going to be used. | |
| **Actor(s):** | Admin | |
| **Preconditions:** | Patient must be in system, Inventory must be in system | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(LastName)       3) LookUp(PatientID)      5) LookUp(ItemName)      7) CreateTreatmentPlan          (             PatientID,             ProcedureID,             InventoryID,             Quantity,             Date          ) | 2)Patients[]  4) Patient:Patients  6) Inventory[]  8)(LOOP)    AddDetails    (        ProcedureID,        InventoryID,        Quantity    )  9)TreatmentPlanNumber |
| **Post-conditions:** | Semester added to system and Id created. | |
| **Outstanding Issues:** |  | |



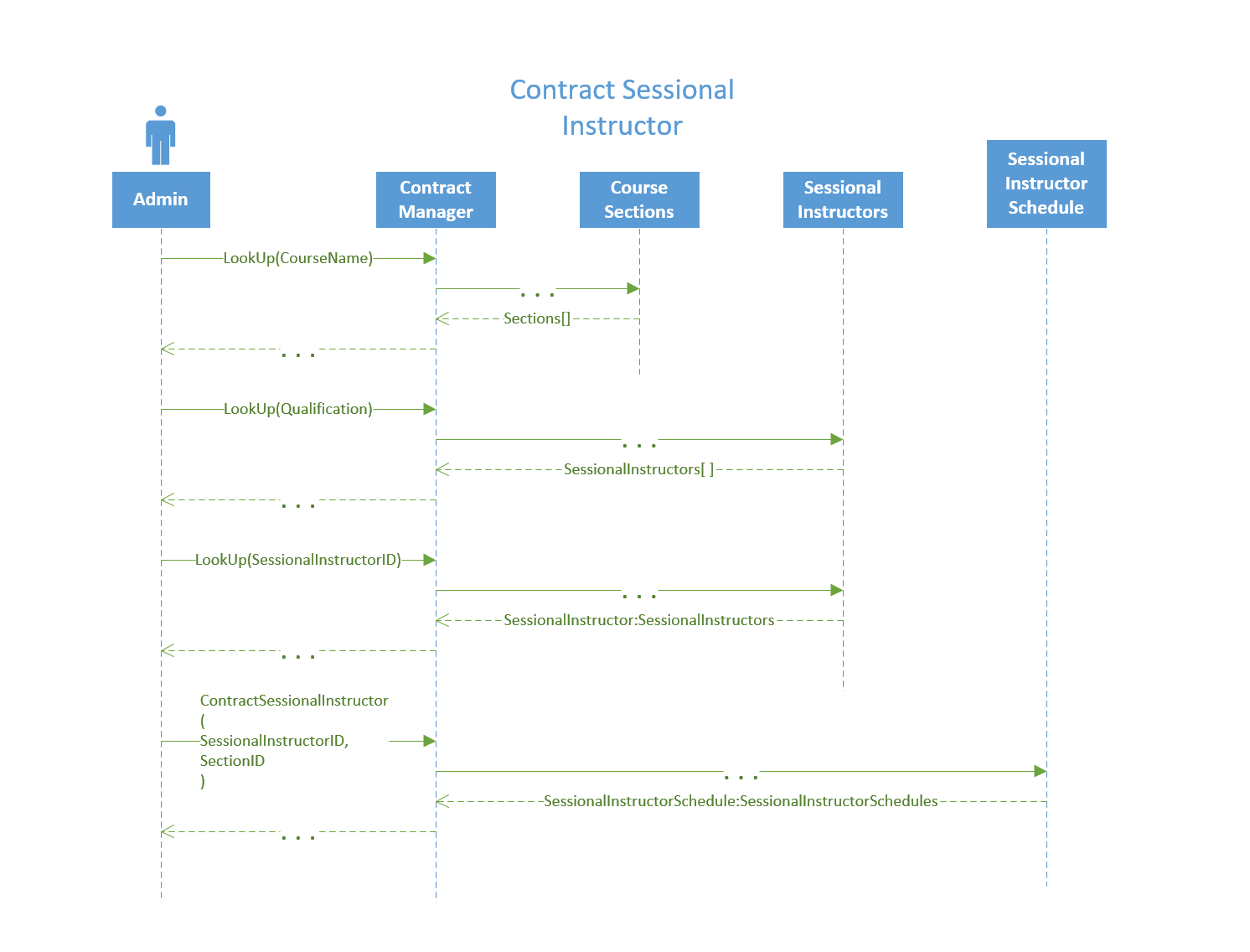
|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Assign Designation | |
| **Description:** | Admin looks up student name and designation name to assign designation to student. | |
| **Actor(s):** | admin | |
| **Preconditions:** | -Student Exists  -Designation Exist | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1) LookUp(StudentName)  3) LookUp(DesignationName)  5) AssignDesignation(              Studentid,              designationId              ) | 2) Students[]  4) Designations[]  5) Confirm |
| **Post-conditions:** | A student is assigned a Designation | |
| **Outstanding Issues:** |  | |



|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Create Schedule | |
| **Description:** | Creates a schedule of class sections for a student | |
| **Actor(s):** | Admin | |
| **Preconditions:** | -Student or Instructor Exists | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1)CreateSchedule  ( StudentID,  SemesterID,  ) | 2)Schedule Created Confirmation returned |
| **Post-conditions:** | Schedule Created Confirmation returned | |
| **Outstanding Issues:** | Can create student schedule, sessional instructor schedule, instructor schedule and classroom schedule. | |



|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Contract Sessional Instructor to Section | |
| **Description:** | Contracts a sessional instructor to a class section | |
| **Actor(s):** | Admin | |
| **Preconditions:** | * Section is created * Sessional Instructor exist | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(CourseName)           3)  LookUp(Designation)      5) LookUp(            SessionalInstructorID         )      7) ContractInstructor(             SessionalInstructorID,             SectionID          ) | 2) Sections[]  4) Instructors[]  6)SessionalInstructor:SessionalInstructors  8)Schedule Update Confirmation |
| **Post-conditions:** | * Instructor assigned to course section | |
| **Outstanding Issues:** |  | |



|  |  |  |
| --- | --- | --- |
| **Use Case Name:** | Look Up | |
| **Description:** | Looks up multiple fields using entered arguments. | |
| **Actor(s):** | Admin | |
| **Preconditions:** |  | |
| **Normal flow of events:** | **Actor action** | **System Response** |
| 1. LookUp(               Arguments           ) | 2) Object() or List<Object> |
| **Post-conditions:** | Depending on if given an ID or alternative info will return either a specific object or a list of objects | |
| **Outstanding Issues:** | * Course (CourseID) or (CourseName) * Designation (Description) * Patient () * Treatment Plan * Class Sections * Inventory * Class Section * Section (CourseName) | |

