**BT MINI CODE**

**Program –**

contract HospitalRecordManagement {

struct PatientRecord {

uint256 recordId;

string patientName;

uint256 admissionTimestamp;

uint256 dischargeTimestamp;

string diagnosis;

string treatment;

string doctor;

}

mapping(uint256 => PatientRecord) public records;

uint256 public recordCount;

event PatientAdmitted(uint256 recordId, string patientName, string diagnosis, string doctor);

event PatientDischarged(uint256 recordId, string patientName, string treatment, string doctor);

function admitPatient(

string memory \_patientName,

string memory \_diagnosis,

string memory \_doctor

) public {

uint256 newRecordId = recordCount++;

PatientRecord storage record = records[newRecordId];

record.recordId = newRecordId;

record.patientName = \_patientName;

record.admissionTimestamp = block.timestamp;

record.diagnosis = \_diagnosis;

record.doctor = \_doctor;

emit PatientAdmitted(newRecordId, \_patientName, \_diagnosis, \_doctor);

}

function dischargePatient(

uint256 \_recordId,

string memory \_treatment

) public {

require(\_recordId < recordCount, "Record does not exist");

PatientRecord storage record = records[\_recordId];

record.dischargeTimestamp = block.timestamp;

record.treatment = \_treatment;

emit PatientDischarged(\_recordId, record.patientName, \_treatment, record.doctor);

}

function getPatientRecord(uint256 \_recordId) public view returns (

uint256,

string memory,

uint256,

uint256,

string memory,

string memory,

string memory

) {

require(\_recordId < recordCount, "Record does not exist");

PatientRecord storage record = records[\_recordId];

return (

record.recordId,

record.patientName,

record.admissionTimestamp,

record.dischargeTimestamp,

record.diagnosis,

record.treatment,

record.doctor

);

}

}