# Project Design Document

## for

# Hospital Management System

**version 1.1**

**Prepared by**
**Group 19**
**Onur Altuğ Akça**
**Onuray Toroslu**
**Gökmen Anıl Dağ**
**Mehmet Ege Bozdemir**
**Berfin Yucak**

**26.12.2022**

# Table of Contents

# Revision History

| Name | Date | Reason for Changes | Version |
|---|---|---|---|
| Project Design Document | 07.11.2022 | Project Design Document is created. | 1.0 |
| Project Design Document | 26.12.2022 | Project Design Document is updated according to feedback. | 1.1 |

# 1. Introduction

This Project Design Document for Hospital Management System provides detailed information on mainly the DBMS of a hospital. The acronyms and abbreviations used in this document are listed in the following subsection. The users that have access to the system, their needs, and corresponding solutions to that needs are discussed in subsection 1.3, Users. The rest of the first section is dedicated to explaining the assumptions and business rules. The HMS is a student project conducted by Group 19 of the CENG 315 Fall semester class of IZTECH and supervised by Associate Professor Belgin Ergenç Bostanoğlu and teaching assistants Büşra Güvenoğlu & Leyla Tekin. Hence, all the assumptions and business rules are based only on case studies.

The second section of the document focuses on providing a clear understanding of the entities and their relations that composes the DBMS via various texts, tables, and schemas.

## 1.1 Definition

The HMS aims to create a medium between the workers and the patients of a hospital while storing needed data for this purpose.The DBMS mainly stores the users' names, phone numbers, addresses, email addresses and passwords; patient records including the admit and discharge dates of patients and their medicines, diagnosis; payment recordings, appointments, and information related to medication.In the light of these, there are some features provided to the users by the system. Some of these features are dependent on 3rd parties due to security concerns. For more information please see also 1.2 Scope.

## 1.2 Scope

The HMS focuses on three different types of users (see also: 1.4 Users) and their needs. The main goal is to manage the hospital personnel, patients, appointments, medication and records with minimum human effort and maximum efficiency and accuracy. The presented features parallel to this goal are listed below:

### 1.2.1 In Scope:

- Classifying and storing departments bu identification number and name, optionally storing the building (block, floor, room) numbers
- Storing the the ID's, names, phone numbers, addresses and emails of all types of personnels
- Assigning personnel to departments
- Classifying the personnels by ID and option to assign different authorization levels to different personnel classes
- Storing any medicine by ID and keeping track of the doses, names and prices of medicines
- Monitoring the prescriptions created/applied by a personnel and received by a patient
- Managing patient records including the dates the patient admitted and discharged if admitted
- Recording given diagnosis
- Making/canceling/rescheduling appointment by choosing date and time
- Storing appointments by ID
- Managing the information about payment status, date and amount

### 1.2.2 Out of Scope:

- Payments ( Only the payment status, amount and if the status is positive the date information are stored in the system. Payment option is not provided with the system, handled by third parties.
- Prescriptions (Prescriptions are handled by TITCK, therefore an entity is not created to represent prescriptions. For more information please see T.R. Ministry of Health e-prescription system, E-Prescription System Integration Document Version 3.1)

## 1.3 Acronyms and Abbreviations

| | |
|---|---|
| DBMS | Database Management System |
| HMS | Hospital Management System |
| CENG | Computer Engineering |
| IZTECH | İzmir Institute of Technology |
| MP | Medical Personnel |
| ER | Entity Relation |
| ERD | Entity Relation Diagram |
| RS | Relational Schema |

## 1.4 Users

The HMS is planned to serve three types of users: Medical Personnels, Patients and Administrators. All three are discussed separately below.

### 1.4.1 Medical Personnels

The MPs refer to the doctors and the nurses that are working at the hospital. Their main needs are foreseen as managing the patient records and the appointments. To be more specific; to view, approve/cancel and add appointments, view the previous diagnoses and medicines that were given to the patient, and add new ones if necessary.

### 1.4.2 Patients

The patients need to manage the same data as the MPs, but on a lower authorization level. A patient can view only their own medical records: Diagnoses, medicines and appointments. The patient can add a new appointment or cancel an upcoming one. But cannot manipulate medicine or diagnoses for security concerns.

### 1.4.3 Administrators

The administrators are the most authorized group in all three types of users. They can manage the personnel or patient data, add/delete departments, remove/add medicines, cancel/approve/transfer appointments. For example, if a personnel is fired or on vacation, a medicine is pulled out of the market, a nurse changes departments etc. administrators are the ones to manipulate the regarding data accordingly.

## 1.5 Assumptions

- The physical infrastructure is provided by the employer.
- All users have sufficient and stable devices and internet connections to access the system.

## 1.6 Business Rules

- Each MP must be assigned to one and only one department.

- Each visit is made with an appointment. Appointments are created by the patients. A patient chooses a department to get an appointment from and chooses an available doctor, date, and time. Unless the appointment is canceled by the doctor or the administrator, it is automatically approved. The patient has the option to cancel it as well.

- For the patient records, the date admitted, ID, the date discharged are entered manually by the personnel at the front desk. The medicine and the diagnosis is entered by the MP. A diagnosis can be done only by a doctor.

- The insurance and other payment details are handled by a third party due to security concerns. In the payment table, only the payment status,amount and date are held to be able to discharge the patient, or present to authorities if needed.

- Every medicine that is given to a patient no matter by whom, must be prescribed by a doctor, and recorded to the patient's file.

- Only a MP can modify a patient record. Patients are only able to view the records.

- A doctor can make the same diagnosis to the same patient or different patients, and a diagnosis can be made by more than one doctor.

- A medicine can be prescribed by more than one doctor, and one doctor can prescribe more than one medicine.

- A medicine can be given to more than one patient, and a patient can take more than one medicine if prescribed.

- A patient can take more than one appointment from a doctor or several appointments from different doctors. But only one patient can be present on one appointment.

# 2. Design

## 2.1 Entity Sets & Attributes

The entity sets and their attributes are listed in separate tables below. The explanatory texts (if needed) follows the concerning table.

| **Personnel** |
| --- |
| **personnelID**<br>personnelName<br>personnelFirstName<br>personnelLastName<br>{personnelPhoneNumber}<br>personnelCounty<br>{personnelEmail}<br>personnelPassword |

➔ All the personnel working at the hospital, such as doctors, nurses, secretaries, officers, administrators, generators, caregivers belong to this entity. They are grouped and distinguished by their ID's. Only some of them which are defined in section 1.3 Users are authorized to login to the system and manipulate data.

| **Patient** |
| --- |
| **patientID**<br>patientName<br>patientFirstName<br>patientLastName<br>{patientPhoneNumber}<br>patientCounty<br>{patientEmail}<br>patientPassword |

➔ All patients are identified with their 11-digits TR ID number.

| **Records** |
| --- |
| **recordID**<br>dateAdmitted<br>dateDischarged |

➔ For each visit, another record is created.

| **Department** |
| --- |
| **departmentID**<br>departmentName<br>building<br>block<br>floor<br>room |

| **Diagnosis** |
| --- |
| **diagnosisID**<br>diagnosisDate<br>patientID |

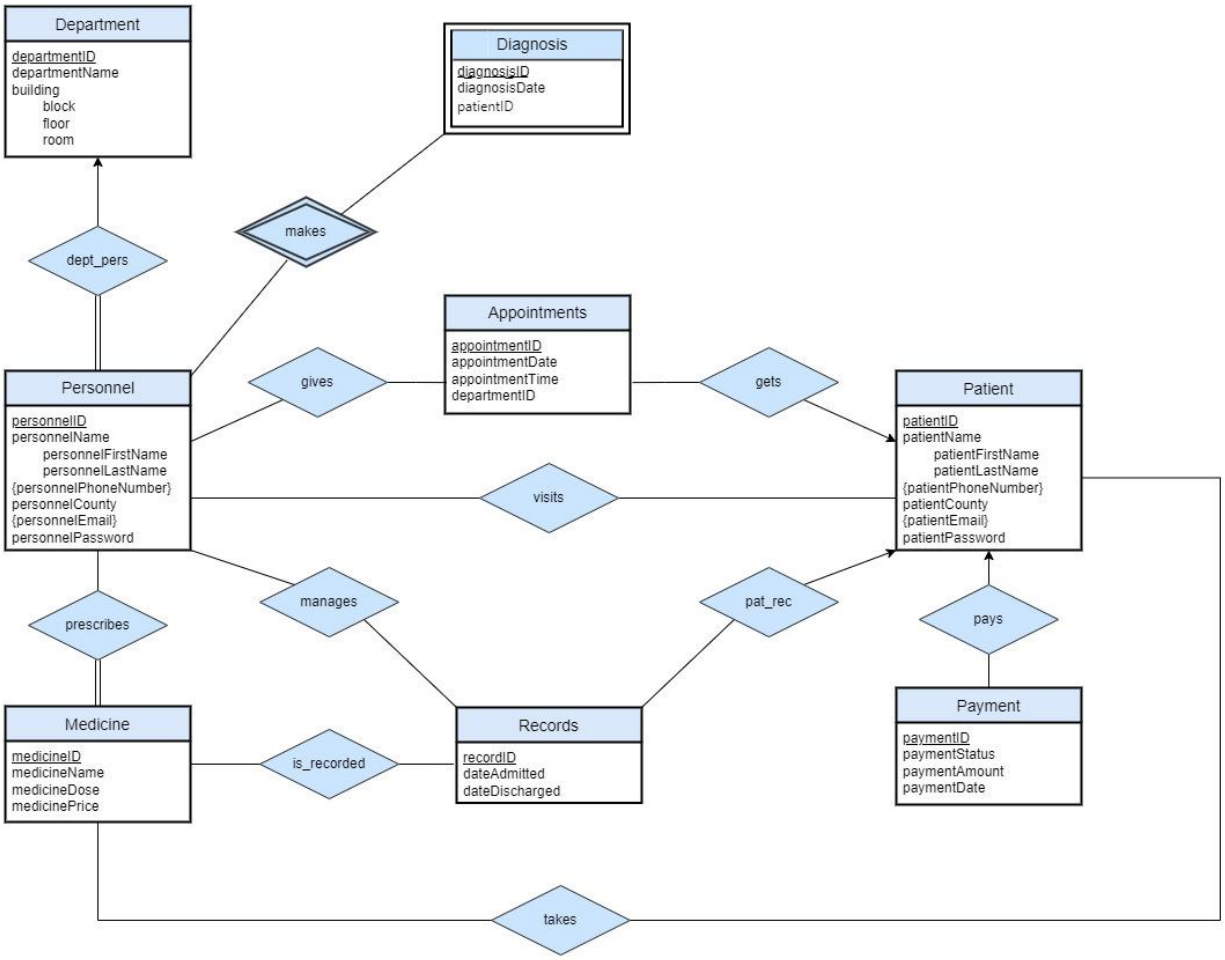
➔ Diagnosis is a weak entity since it can only be made by a doctor and does not exist otherwise.

| **Payment** |
| --- |
| **paymentID**<br>paymentStatus<br>paymentAmount<br>paymentDate |

| **Medicine** |
| --- |
| **medicineID**<br>medicineName<br>medicineDose<br>medicinePrice |

| **Appointments** |
| --- |
| **appointmentID**<br>appointmentDate<br>appointmentTime<br>departmentID |

## 2.2 ER Diagram




## 2.3 Relation Sets & Attributes

To create the relational schema, the necessary steps for converting the ERD to RS by reduction are listed and explained below.

### 2.3.1 Step 1: Listing the Entities

Department (<u>departmentID</u>, departmentName,
block, floor, room)

Personnel (<u>personnelID</u>,
personnelFirstName, personnelLastName,
personnelCounty,
personnelPassword)

- Since phone number and email are multivalued attributes, they are represented by separate schemas:

Personnel_PhoneNumber ( personnelID, personnelPhoneNumber )

Personnel_Email ( personnelID, personnelEmail )

Medicine (medicineID, medicineName, medicineDose, medicinePrice)

Records (recordID, dateAdmitted, dateDischarged)

Payment (paymentID, paymentStatus, paymentAmount, paymentDate)

Patient (patientID,
patientFirstName, patientLastName,
patientCounty,
patientPassword)

- Since phone number and email are multivalued attributes, they are represented by separate schemas:

Patient_PhoneNumber ( patientID, patientPhoneNumber )

Patient_Email ( patientID, patientEmail )

Appointments (appointmentID, appointmentDate, appointmentTime, departmentID)

Diagnosis (personnelID, diagnosisID, diagnosisDate, patientID)

- Since Diagnosis is a weak entity, the primary key of the identifying strong entity set Personnel is included.

### 2.3.2 Step 2: Listing the Relations

- Between the entities Personnel & Department:

dept_pers = (personnelID, departmentID)

- Between the entities Records & Patient:

pat_rec = (recordID, patientID)

- Between the entities Personnel & Medicine:

prescribes = (personnelID, medicineID)

- Between the entities Personnel & Appointments:

gives = (personnelID, appointmentID)

- Between the entities Personnel & Patient:

visits = (personnelID, patientID)

- Between the entities Personnel & Records:

manages = (personnelID, recordID)

- Between the entities Medicine & Records:

is_recorded = (medicineID, recordID)

- Between the entities Patient & Payment:

pays = (patientID, paymentID)

- Between the entities Patient & Appointments:

gets = (patientID, appointmentID)

- Between the entities Patient & Medicine:

takes = (patientID, medicineID)

Since the diagnose relation links the weak entity set Diagnosis to its identifying strong entity set Personnel and the Diagnosis schema already contains the attributes that would appear in the diagnose schema, the relationship set between the entities Diagnosis and Personnel is not included to this list.

### 2.3.3 Step 3: Redundancy of Schema

Department (departmentID, departmentName,
block, floor, room)

Personnel (personnelID,
personnelFirstName, personnelLastName,
personnelCounty,
personnelPassword,
**departmentID**)

- departmentID referencing the relation between the entities Department and Personnel. Since the relationship is total on the many side, the primary key of the Department is added to the Personnel.

Personnel_PhoneNumber ( personnelID, personnelPhoneNumber )

Personnel_Email ( personnelID, personnelEmail )

Medicine (medicineID, medicineName, medicineDose, medicinePrice)

Records (recordID, dateAdmitted, dateDischarged, **patientID**)

- patientID referencing the relation between the entities Patient and Records. The two schemas are combined even though the participation is partial by using null values.

Payment (paymentID, paymentStatus, paymentAmount, paymentDate, **patientID**)

- patientID referencing the relation between the entities Patient and Payment.The two schemas are combined even though the participation is partial by using null values.

Patient (patientID,
patientFirstName, patientLastName,
patientCounty,
patientPassword)

Patient_PhoneNumber ( patientID, patientPhoneNumber )

Patient_Email ( patientID, patientEmail )

Appointments (appointmentID, appointmentDate, appointmentTime, departmentID, **patientID**)

- patientID referencing the relation between the entities Patient and Appointments.The two schemas are combined even though the participation is partial by using null values.

Diagnosis (personnelID, diagnosisID, diagnosisDate, patientID)

~~dept_pers =(personnelID, departmentID)~~

~~pat_rec =(recordID, patientID)~~

- Since pat_rec is a many-to-one relationship set, it is represented as a schema with extra attribute to the “many” side, containing the primary key of the “one” side. The two schemas are combined even though the participation is partial by using null values.

prescribes = (personnelID, medicineID)

- Since prescribes is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Personnel and Medicine.

gives = (personnelID, appointmentID)

- Since gives is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Personnel and Appointments.

visits = (personnelID, patientID)

- Since visits is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Personnel and Patient.

manages = (personnelID, recordID)

- Since manages is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Personnel and Records.

is_recorded = (medicineID, recordID)

- Since is_recorded is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Medicine and Record.

~~pays =(patientID, paymentID)~~

- Since pays is a many-to-one relationship set, it is represented as a schema with extra attribute to the “many” side, containing the primary key of the “one” side. The two schemas are combined even though the participation is partial by using null values.

~~gets =(patientID, appointmentID)~~

- Since gets is a many-to-one relationship set, it is represented as a schema with extra attribute to the “many” side, containing the primary key of the “one” side. The two schemas are combined even though the participation is partial by using null values.

takes = (patientID, medicineID)

- Since takes is a many-to-many relationship set, it is represented as a schema with attributes for the primary keys of the two participating entity sets: Patient and Medicine.

Notes:

Underlined words refer to the primary keys.
Bold words refer to the foreign keys.
Overlined words refer to the redundant relations.

## 2.4 Relational Schema

Department (departmentID, departmentName,
block, floor, room)

Personnel (personnelID,
personnelFirstName, personnelLastName,
personnelCounty,
personnelPassword,
**departmentID**)

Personnel_PhoneNumber ( personnelID, personnelPhoneNumber )

Personnel_Email ( personnelID, personnelEmail )

Medicine (medicineID, medicineName, medicineDose, medicinePrice)

Records (recordID, dateAdmitted, dateDischarged, **patientID**)

Payment (paymentID, paymentStatus, paymentAmount, paymentDate, **patientID**)

Patient (patientID,
patientFirstName, patientLastName,
patientCounty,
patientPassword)

Patient_PhoneNumber ( patientID, patientPhoneNumber )

Patient_Email ( patientID, patientEmail )

Appointments (appointmentID, appointmentDate, appointmentTime, departmentID, **patientID**)

Diagnosis (personnelID, diagnosisID, diagnosisDate, patientID)

prescribes = (personnelID, medicineID)

gives = (personnelID, appointmentID)

visits = (personnelID, patientID)

manages = (personnelID, recordID)

is_recorded = (medicineID, recordID)

takes = (patientID, medicineID)