DB Design

Patient: Stores the data of all the patients registered with the application.

Attributes: PatientID, Username, Password, DoctorID (Doctor), SectionsCompleted, EmailVerified, Name, Age, Sex, Email, Address, DOB, Marital Status and other relevant information.

Specialist: Stores the data of the Specialists registered.

Attributes: SpecID, Username, Password, Name, Email.

Doctor: Stores the data of the Doctors.

Attributes: DoctorID, Username, Password, Name, Email, SpecialistID (Specialist), RoomCode.

Admin: Stores the details of the Admins.

Attributes: AdminID, Username, Password, Name, Email.

Legend:

----- : Primary Key

----- : Surrogate Key

----- : Foreign Key

----- : Unique

API Design

/login: Allows the user (Patient, Doctor, Admin or Specialist) to login to the application.

Method: POST

Request Body:

username – Username of the user.

password – Password of the user.

type – Doctor, Specialist, Patient or Admin.

Returns: A Status object stating if the transaction was executed properly or not.

/register: Allows registration of only Patients.

Method: POST

Request Body:

Username – Username selected by the user.

Password - Password of the user.

DoctorID (Doctor) – Optionally the user can select the doctor assigned to him/her.

Name – Name of the Patient.

Email – Email of the user.

Age – Age of the Patient.

Sex – Sex of the Patient.

Address – Address of the Patient.

DOB – DOB of the Patient.

Marital Status – Marital Status of the Patient.

Other Relevant Fields.

Returns: A Status object stating if the transaction was executed properly or not.

/findExistingEmail/{email}: Checks if the email is already in database.

Method: GET

Attributes: email to be checked.

Returns: A Status object stating if the email exists in the database or not.

/findExistingUName/{username}: Checks if the username is already in database.

Method: GET

Attributes: username to be checked.

Returns: A Status object stating if the username exists in the database or not.

{username}/progress/{section}: Updates the Section wise progress of the patient through the material.

Method: GET

Attributes: username of the user who's logged in.

section which the Patient most recently visited.

Returns: A Status object stating if the transaction was executed properly or not.

/updateProfile: User can update its own profile. However, some details can't be updated by the user such as medical records.

Method: POST

Request Body:

Name – Name of the Patient.

Age – Age of the Patient.

Address – Address of the Patient.

Marital Status – Marital Status of the Patient.

Other Relevant Fields.

Returns: A Status object stating if the transaction was executed properly or not.

/forgotPassword: User can update its password after authentication.

Method: POST

Request Body:

password: Old Password

new-password: New password

Returns: A Status object stating if the transaction was executed properly or not.

/logout: Logs the user out and clears any relevant session variables.

Method: POST

Returns: A Status object stating if the transaction was executed properly or not.

/progress/{username}: Doctors can check the Progress of a particular user.

Method: POST

Attributes: username of the patient whose progress is to be checked.

Returns: A Status object which contains the sections completed by the Patient.

/generateRoomCode/{username}: Generates the room code of a Doctor.

Method: POST

Attributes: username of the doctor.

Returns: A Status object stating if the transaction was executed properly or not.

/joinRoom: Joins any existing room of a doctor.

Method: POST

Request Body:

room-code: Room Code provided by the doctor to the patient.

Returns: A Status object stating if the transaction was executed properly or not.

/progress/{username}: Doctors can check the Progress of a particular user.

Method: POST

Attributes: username of the patient whose progress is to be checked.

Returns: A Status object which contains the sections completed by the Patient.

/remSections/{username}/{sectionList}: Removes access of the sections from a particular user.

Method: POST

Attributes: username of the patient access is to be edited.

sectionList: list of sections which are to be removed.

Returns: A Status object stating if the transaction was executed properly or not.

/addSections/{username}/{sectionList}: Removes access of the sections from a particular user.

Method: POST

Attributes: username of the patient access is to be edited.

sectionList: list of sections which are to be added.

Returns: A Status object stating if the transaction was executed properly or not.

/editPatient/{username}: Edits all the data of the Patient.

Method: POST

Attributes: username of the patient access is to be edited.

Request Body:

DoctorID (Doctor) – Optionally a Doctor's ID can be given.

SectionsCompleted – Sections completed by the User.

Name – Name of the Patient.

Age – Age of the Patient.

Sex – Sex of the Patient.

Address – Address of the Patient.

DOB – DOB of the Patient.

Marital Status – Marital Status of the patient.

Other Relevant Fields.

Returns: A Status object stating if the transaction was executed properly or not.

/delPatient/{username}: Deletes a Patient.

Method: POST

Attributes: username of the patient access is to be edited.

Returns: A Status object stating if the transaction was executed properly or not.

/registerDoc: Register a new Doctor in the system.

Method: POST

Request Body:

username – Username of the Doctor.

password – Password

name – Name of the Doctor.

email – Email of the Doctor.

specialistID (Specialist) – Specialist responsible for the Doctor.

Returns: A Status object stating if the transaction was executed properly or not.

/editDoc/{username}: Edits the data of a Doctor.

Method: POST

Attributes: username of the patient access is to be edited.

Request Body:

username – Username of the Doctor.

password – Password

name – Name of the Doctor.

email – Email of the Doctor.

specialistID (Specialist) – Specialist responsible for the Doctor.

Returns: A Status object stating if the transaction was executed properly or not.

/delDoc/{username}: Deletes a Doctor.

Method: POST

Attributes: username of the doctor to be deleted.

Returns: A Status object stating if the transaction was executed properly or not.

/registerSpec: Register a new Specialist.

Method: POST

Request Body:

username – Username of the Specialist.

password - Password

name – Name of the Specialist.

email – Email of the Specialist.

/editSpec/{username}: Edits the details of a Specialist.

Method: POST

Attributes: username of the patient access is to be edited.

Request Body:

username – Username of the Specialist.

password - Password

name – Name of the Specialist.

email – Email of the Specialist.

Returns: A Status object stating if the transaction was executed properly or not.

/delSpec/{username}: Deletes a Specialist.

Method: POST

Attributes: username of the specialist to be deleted.

Returns: A Status object stating if the transaction was executed properly or not.