

Jan 6 , 2025

SE 4458 Software Architecture & Design of Modern Large Scale Systems - Final

You are assigned to one of projects below for final assignment. Please email me for any questions you have on requirements

Group 1 : Create a Doctor appointment site

Create a doctor appointment system (like doktortakvimi.com). Implement both frontend UIs and backend services via APIs.

Below are use cases for functional and non-functional requirements.

DOCTOR : REGISTERS

ADD ME AS DOCTOR

email: doctor@gmail.com Authenticate with Google

Fullname:

Area of Interest: Orthopedics, Pediatrics...

Available Days: ☐ Mon ☐ Tue ☐ Wed ☐ Thu ☐ Fri ☐ Sat ☐ Sun

Available Hours: 10:00 - 20:00

Adress:

City: Istanbul, Ankara, Izmir

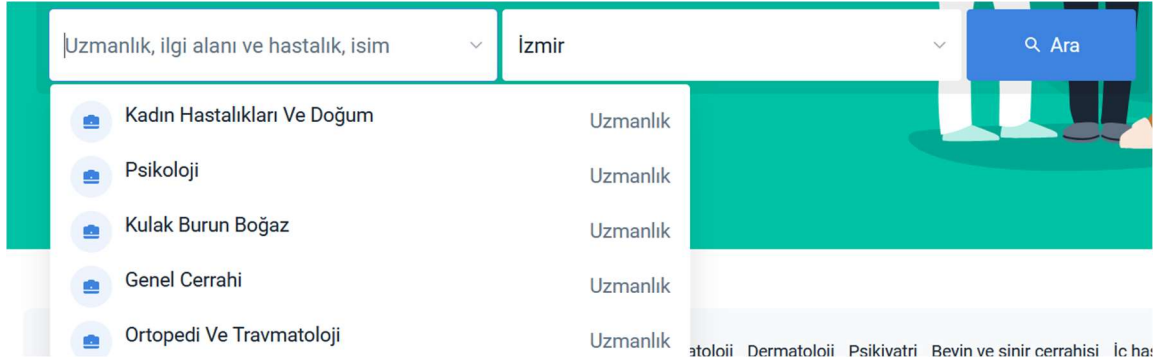
Town: ComboBox REGISTER

Doctors will give extra information after google authentication

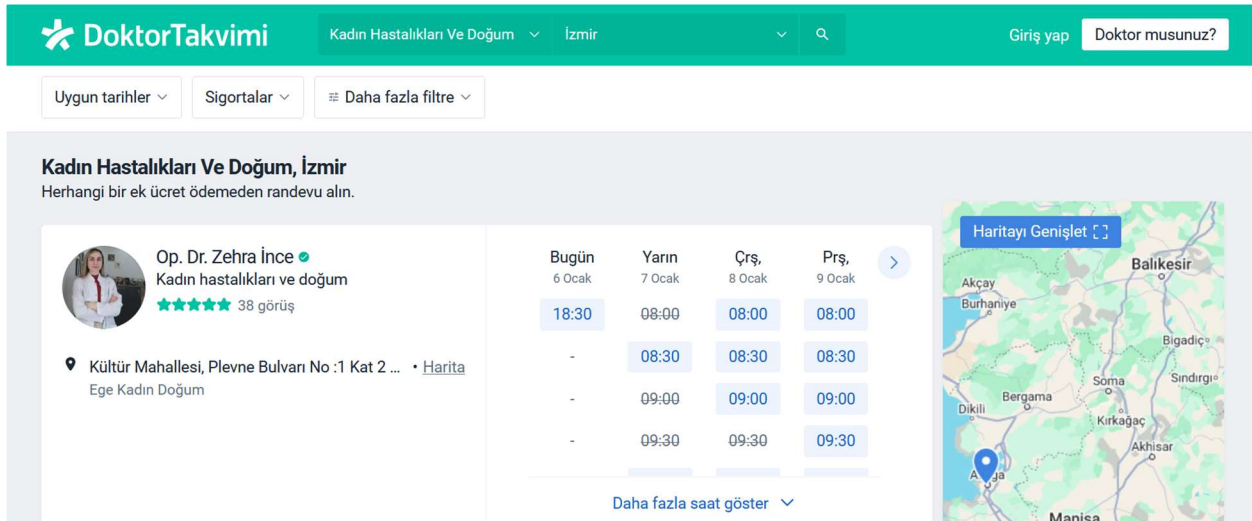
- On home screen, users can register as a doctor. They have to login with Google after which they will details about their availability.
- Address details need to be precisely taken for later usage with maps
- An authorized admin will be able to approve registrations after contacting doctor and related organizations. They will enter exact address of doctor's office for later map usage

PATIENT : SEARCHES DOCTOR

Search



Results



Kadın Hastalıkları Ve Doğum, İzmir
Herhangi bir ek ücret ödmeden randevu alın.

Op. Dr. Zehra İnce
Kadın hastalıkları ve doğum
★★★★★ 38 görüş

Kültür Mahallesi, Plevne Bulvarı No :1 Kat 2 ... • [Harita](#)
Ege Kadın Doğum

Bugün 6 Ocak	Yarın 7 Ocak	Çrş. 8 Ocak	Prş. 9 Ocak
18:30	08:00	08:00	08:00
-	08:30	08:30	08:30
-	09:00	09:00	09:00
-	09:30	09:30	09:30

[Daha fazla saat göster](#)

- Patients can search a doctor of a certain interest area. All matching doctors should be displayed on a map
- Doctor search should autocomplete filtering on name, area
- All available times of doctor should be shown for CURRENT week

PATIENT : MAKES AN APPOINTMENT

- Customer makes an appointment by clicking
- They will need to sign in with Google for later account and appointment management
- Customer might not finish an appointment. See next use case on how to handle this

PATIENT : REVIEWS THE DOCTOR

- After each visit, patient gets an email notification to rate the doctor. Create a notification to send to patient to rate the doctor with a comment. Check the inappropriate language and decline rating when necessary

Please rate your visit to doctor [Zehra Ince](#)



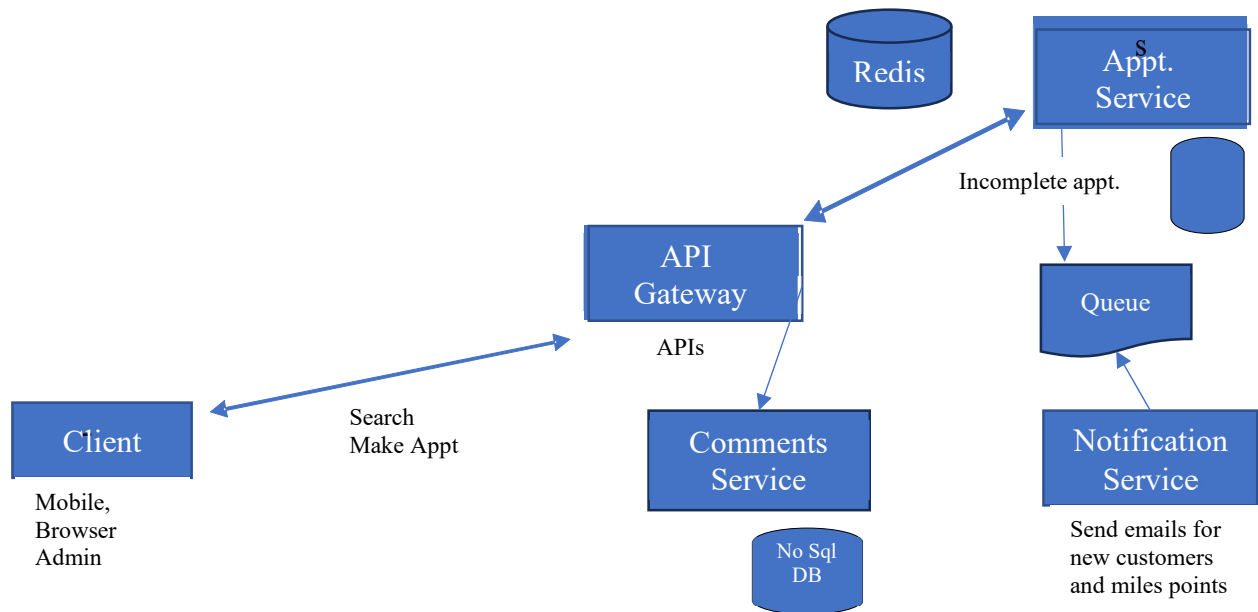
Comments

Kesinlikle tavsiye ediyorum, bacağıma hissetmiyorken girdiğim ameliyattan sadece 3 saat sonra ayağa kalkıp yürüdüm. Hem ilgili hem işinde son derece başarılı bir cerrah.

SYSTEM : SENDS NOTIFICATIONS

- INCOMPLETE APPOINTMENT REQUEST NOTIFICATION : When a patients searches for a doctor and goes into details, signs in but does NOT make an appointment, the system send a daily notification to those user if they want to finish their appointment
 - o For this, you will utilize a queue whenever such event occurs. You will create a scheduled task that reads from this queue as this can happen 1000s of times a day
-
- For email send, you can use a gmail account see <https://kinsta.com/blog/gmail-smtp-server/>

NON-FUNCTIONAL REQUIREMENTS



- Patient comments will be stored in a No Sql DB. You can use any Cloud Service for this purpose.

Students

Melda Yazgin Pelin Sude Kirli Eftelya Sivridag Kutay Mehmet Cetiner	Emirhan Kursun Aytun Yuksek Mustafa Mert Kilinc Murat Ege Baykent
ÖMER DOĞAN İBRAHİM ÇELİK CEM ARDA DOĞAN MERTCAN VURAL Volkan Ege Kilinc	HALİM AVCI ABDULLAH ASLAN GÖRKEM ERTAŞ EGE ÖZLEM SELİN AKBAY

Group 2: Create a Prescription and Doctor Visit Management System

Create a prescription and doctor visit management system for Saglik Bakanligi for pharmacies. Implement Below are use cases for functional and non-functional requirements.

DOCTOR : CREATES PRESCRIPTION FOR PATIENT

PRESCRIPTION FOR PATIENT

Prescription Id: 123456

TC Id no: 12345678901

Fullname: Mert Can

Medicine:

Name
Norodol 3X1
Voltaren 1X1
Nurofen 3X1

- Doctor creates a prescription for a patient after the visit
- The medicines will be a no-authentication lookup which is explained in next use cases.
- Each visit is recorded in the system
- TC lookup can be built by a mock API service like <https://mocki.io/> , you don't need build an API service
- You can create a constant user id/password for doctors. Create prescription services must be authorized. You can use JWT authentication for this

PHARMACY : SUBMITS PRESCRIPTION FOR PATIENT

Pharmacy: GUNEY Eczane

TC Id no: 12345678901

Fullname: Mert Can

Prescription Id: 123456

Medicine:

Name	Price
Norodol	20
Naprosyl	50

Total : 70 TL

- Will be used by pharmacies to enter prescriptions
- Pharmacies will need to authenticate by to submit a prescription
- Medicine lookup is explained in next use cases. Please add random prices for medicines.
- You will utilize a queue when a prescription is incomplete
- TC lookup can be built by a mock API service like <https://mocki.io/> , you don't need build an API service
- Medicine search should autocomplete
- You can create a constant user id/password for doctors. Submit prescription services must be authorized

SYSTEM : POPULATES MEDICINE NAMES

- There is no medicine lookup service by Saglik Bakanligi
- Medicine lists are published weekly at <https://www.titck.gov.tr/dinamikmodul/43> .
- You will write a no-authentication REST API that will enable querying medicine name
- i.e if you search any medicine that ASP word in it, it will return

```
{"medicationNames":  
  ["ASPIRIN","CASPOBIEM","CASPOPOL","CORASPIN","SIGMASPORIN","VAS  
  PARIN"]  
}
```

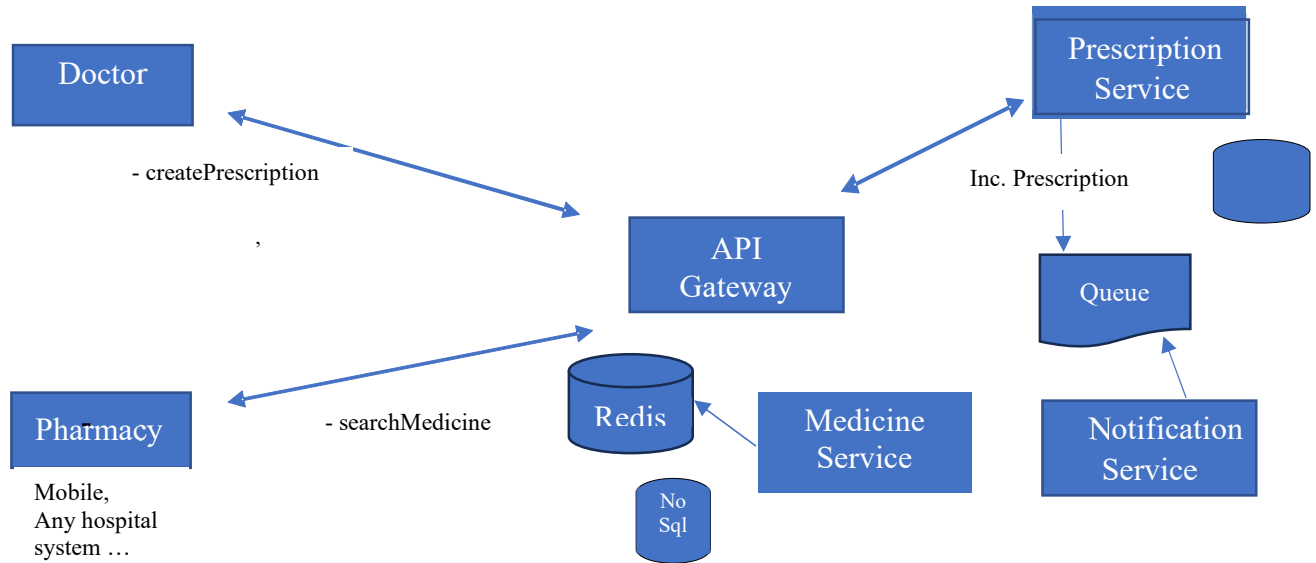
- you will need to download latest file and write an Excel parsing program to upload this to your database. The medicine data needs to be refreshed every week on Sunday 22.00. You can keep history of medicine list or create a logic to keep the ones that have been used in prescriptions
- doing this via a mockup service like <https://run.mocky.io> IS NOT ACCEPTABLE, data must reside in your data source

SYSTEM : NOTIFIES INCOMPLETE PRESCRIPTIONS TO PHARMACIES

- A separate notification service will be developed that will browse all prescriptions submitted at current date and will email a report to each pharmacy for the prescriptions with missing medicine. This will run at 1:00 every night

To: GUNEY Eczane
You have 10 incomplete prescriptions today
Pres Id : 12345 Missing Voltaren

NON-FUNCTIONAL REQUIREMENTS



- Medicine data will be stored in a No Sql DB

Students

Mert Erisir Babur Gur Alper Han Uyanik Birkan Cemil Abaci	Baris Nisanci HANDE HAZAN KESKİN Efe Ilhan Ata Ekren
TURGUT EGEMEN ÖZYÜREK POYRAZ SİVRİKAYA GİRAY AKSAKAL DAĞCA ÜNÜVAR Mert Kahraman	İRFAN EMRE UTKAN LEVENT KILIÇ İBRAHİM HALİL ÖZDEMİR ALİ ALP KÜREL

COMMON REQUIREMENTS

- Project can be built on any service-oriented framework as long as requirements are met.
- Simple UI implementation per mock-ups given above is required. Front end UIs do not have to be same as shown above. It just needs to work
- All business use cases above must be available via APIS. Could be REST, GraphQL etc. All APIs must be versionable and support pagination when needed. All APIs will need to be reached via an API gateway
- You can use RabbitMQ or Azure Messaging for queue solutions. If you cant implement it, you will need to implement it synchronously but you will lose points
- At least one caching solution needs to be implemented using Redis or any other caching solution i.e City and Town names for addresses. Medicine names
- You can make assumptions as long as you document them
- The project will be deployed to providers like Azure, Google Cloud, Render and AWS or any other cloud vendor
- MAKE SURE TO CREATE A DOCKER FILE in your source. See an example at <https://www.digitalocean.com/community/tutorials/how-to-build-a-node-js-application-with-docker-for-a-node.js> application. DO NOT CREATE A DOCKER IMAGE FILE, it would be too big and not needed
- DEPLOYMENT
 - o create a data model for each data source and use a DATABASE service from any cloud service you like i.e Azure SQL Server, Azure Cosmos Table. SQLite is NOT allowed
 - o For API and UI layer hosting of app services, use a cloud service i.e Azure App Services
 - o For API gateway, you can use the gateway you implemented in assignment. Azure API management is costly so refrain from using it unless you want to pay for it
 - o For queue service, use Azure Message Queues, Rabbit MQ
 - o For scheduling services, use a cloud service i.e Azure App Logic or <https://cloud.google.com/scheduler/>

DELIVERABLES

- Public Github repository link
- A README document in your GITHUB repository that has
 - o Final deployed urls of the application
 - o your design, assumptions, and issues you encountered.
 - o Data models (i.e an ER)
 - o Include a link to a short video (max 5 minutes) presenting your project.