Journal of Information Systems and Management Research
Bilişim Sistemleri ve Yönetim Araştırmaları Dergisi

http://dergipark.gov.tr/jismar

Araştırma Makalesi / Research Article

Fakülte Öğrenci İşleri Birimi İçin MEAN Yığını Kullanılarak Destek Yönetim Sistemi Geliştirilmesi

İlknur TEKE*[a] Çiğdem TARHAN[b],

[a]*Dokuz Eylül Üniversitesi, İktisadi İdari Bilimler Fakültesi, Yönetim Bilişim Sistemleri Bölümü, İZMİR, 35340, TÜRKİYE*
[b]*Dokuz Eylül Üniversitesi, İktisadi İdari Bilimler Fakültesi, Yönetim Bilişim Sistemleri Bölümü – Bölgesel Kalkınma ve İşletme Bilimleri Araştırma ve Uygulama Merkezi, İZMİR, 35340, TÜRKİYE*

MAKALE BİLGİSİ

Alınma: 09.05.2023
Kabul: 28.06.2023

Anahtar Kelimeler:
Öğrenci İşleri, Dijitalleşme, Destek Yönetim Sistemi, MEAN Yığını

Sorumlu Yazar
e-posta:
ilknuryldrm@gmail.com

ÖZET

Endüstri 4.0 kavramının ortaya çıkmasıyla dijitalleşme trendleri gerek kamu kuruluşlarında gerekse özel sektörde iş süreçlerinin değişmesine neden olmaktadır. İş süreçlerinin dijital ortama aktarılmasını içeren dijitalleşme çalışmaları yükseköğretim kurumları için de önemli bir role sahip olmuştur. Bu makalede Dokuz Eylül Üniversitesi İktisadi ve İdari Bilimler Fakültesi bünyesinde bulunan öğrenci işleri biriminin öğrencilerle iletişiminde herhangi bir yazılım kullanılmaması, öğrencilerin danışmak istedikleri konularla ilgili fiziksel başvuruda bulunması çalışmanın problemi olarak belirlenmiştir. Bu bağlamda birimin kullanımına yönelik destek yönetim sistemi geliştirilmiştir. JavaScript tabanlı MEAN (MongoDB, Express.js, Angular, Node.js) yığını kullanılarak geliştirilen sistem içerisinde kategorize edilmiş sıkça sorulan sorular yer almaktadır. Soruların yetersiz gelmesi durumunda ise öğrencinin bölüm bilgisi ile eşleşen birim personeline göndermek üzere danışmak istediği konu ile ilgili yazılı talep oluşturma işlemi bulunmaktadır. Bunlarla beraber kullanıcı doğrulama işlevi ile veritabanında öğrencinin güncel iletişim bilgilerinin bulunması amaçlanmaktadır. Bu sistem ile öğrenci işleri biriminin öğrencilerle iletişimini iyileştirmesi ve çalışmalarını daha verimli hale getirmesi beklenmektedir.

DOI: 10.59940/jismar.1294834

Development of a Sussort Management System for Faculty Student Affairs Unit Using MEAN Stack

ARTICLE INFO

Received: 09.05.2023
Accepted: 28.06.2023

Keywords:
Student Affairs, Digitalization, Sussort Management System, MEAN Stack

Corresponding Authors
e-mail:
ilknuryldrm@gmail.com

ABSTRACT

With the concept of Industry 4.0, digitalisation trends cause business processes to change. Digitalisation processes, which include the transfer of business processes to the digital environment, have also played an important role for higher education institutions. In this article, it is determined as the problem of the study that the student affairs unit within the Faculty of Economics and Administrative Sciences of Dokuz Eylül University does not use any software in communication with students and students physically assly for the issues they want to consult. In this context, a sussort management system was developed for the student affairs unit. The system, developed with JavaScript-based MEAN (MongoDB, Express.js, Angular, Node.js) stack, includes categorised frequently asked questions. In case the questions are insufficient, there is a process of creating a written request about the subject the student wants to consult. It is sent to the unit staff matching the student's department information. In addition, it is aimed to have the current contact information of the student in the database with user verification. With this system, it is expected that the student affairs unit will improve its communication with students and make its work more efficient.

DOI: 10.59940/jismar.1294834

1. GİRİŞ *(INTRODUCTION)*

Günümüzde, endüstri 4.0 kavramı ve dijitalleşme trendleri gerek kamu kuruluşlarında gerekse özel sektörde iş süreçlerinin değişmesine neden olmaktadır. Bu bağlamda, dijitalleşme trendlerinin etkilediği sektörlerdeki kurum ve kuruluşlarda atılması gereken ilk adımın fiziksel olarak yürütülen süreçlerin elektronik ortama taşınması olarak belirlenmiştir. Dijital dönüşüm olarak da adlandırılan dijitalleşme, kurum ve kuruluşlardaki iş süreçlerini yürüten kişilerin bu süreçleri dijital teknolojileri kullanarak kolaylaştırması olarak tanımlanmaktadır [1]. Dijital çağın dinamiklerinin getirisi olarak yükseköğretim kurumları da teknolojik gelişmelerde yaşanan hızlı değişimler nedeniyle farklı eğilimler gösteren öğrenci gruplarıyla nasıl daha etkili bir eğitim sağlayacaklarını düşünmek zorunda kalmışlardır [2]. Yükseköğretim kurumları, öğrencilere iş dünyasının ihtiyaçlarına uygun olarak öğretim vermek için iş süreçlerini yenilemek ve teknolojik gelişmelere ayak uydurma çabası içerisindedir. Bu nedenle, ihtiyaçların zaman-mekân bağımsız olarak giderilmesini amaçlayan dijital dönüşüm trendleri, yükseköğretim kurumlarındaki iş süreçlerinin yeniden düzenlenmesinde ve öğrencilerin dijital çağa hazırlanmasında önemli bir rol oynamaktadır. Üniversitelerin bünyesinde yürütülen süreçler temel olarak idari ve akademik olmak üzere ikiye ayrılmaktadır. İdari süreçler kurumun yönetimiyle ilgili fonksiyonlardan oluşurken, akademik süreçler öğretim fonksiyonlarından oluşmaktadır. Bir üniversite için dijitalleşme kavramı sadece öğretim faaliyetlerinin dijital ortamda gerçekleşmesi değil, kurumun tüm idari süreçlerinin de dijital ortamda yürütülmesi anlamına gelmelidir [3].

Ülkemizde bulunan Yükseköğretim Kurumlarının idari teşkilat şemasında yer alan Öğrenci İşleri Daire Başkanlığı, öğrencilerin kayıt, kabul ve ders durumları ile ilgili işlemler ile mezuniyet, kimlik, burs, mezunların izlenmesi ve benzeri diğer görevleri yerine getirmekle görevlidir [4]. Bu görevler esas niteliği taşırken, öğrenci işleri daire başkanlığının danışmak ya da sormak istediği konularla ilgili iletişim kuran öğrenciye destek vermesi de birim için ciddi bir iş yükü oluşturmaktadır. Bu makalede, Dokuz Eylül Üniversitesi (DEÜ) Öğrenci İşleri Daire Başkanlığına bağlı olarak görev yapan İktisadi ve İdari Bilimler Fakültesi (İİBF) öğrenci işleri birimi çalışma alanı olarak seçilmiştir. Bu birime danışmak isteyen öğrencilere yönelik bir yazılım bulunmadığı için fiziksel başvuru yapılması, bu başvuruların takip edilememesi, tekrarlayan nitelikte olması ve bu süreçlerin neden olduğu iş yükünün tespit edilememesi çalışmanın problemi olarak tanımlanmıştır. Öğrencilerin danışmak ya da sormak istediği konu ile ilgili kullanabileceği, öğrenci işleri biriminin öğrencilerle arasındaki iletişimi dijital ortama taşıyarak güçlendirecek, bir uygulama geliştirilmesi çalışmanın amacı olarak belirlenmiştir.

1.1. Problem Tanımı *(Problem Definition)*

DEÜ-İİBF bünyesinde bulunan öğrenci işleri birimi ile nitel veri toplama yöntemi olan odak görüşme gerçekleştirilmiştir. Yapılan görüşme sonrası tespit edilen problemler teknik nedenlerden kaynaklanan altyapı problemleri, öğrenciler ile birim arasında yaşanan iletişim problemleri ve idari problemler olmak üzere 3 ana başlıkta toplanmıştır.

1. Altyapı Kaynaklı Problemler: Kurumda öğrenci işleri süreçlerine yönelik kullanılan yazılımlar nedeni ile yaşanan teknik problemlerdir. Bu konuda öne çıkan en önemli sorun mevcutta kullanılan yazılımların birbirinden bağımsız olarak çalışmasından kaynaklanan aynı verilerin farklı formatlarda tutulmasıdır.

2. İletişim Problemleri: Öğrencilere iletişim bilgilerinin yetersizliğinden kaynaklı gerekli bilgi akışının sağlanamaması ve öğrencinin idari süreçlerle ilgili ihtiyaçları için fiziksel başvuruda bulunması bu konuda en önemli problemler olarak görülmektedir.

3. İdari Problemler: Öğrenci işleri birimi içerisindeki adam/saat verimliliğini maksimize etme çabasından kaynaklanan problemlerdir.

Bu makalede, iletişim problemleri başlığı altında toplanan problemlerin öğrenci işleri biriminde yürütülen iş süreçlerine etkileri üzerinde durulmuştur. Bu problemlere çözüm olması adına uygulanabilecek yenilikçi yöntemler araştırılmış ve çalışma kapsamında sıkça sorulan soruları da içeren destek yönetim sistemi adıyla JavaScript tabanlı bir web uygulaması geliştirilmiştir.

2. LİTERATÜR TARAMASI *(LITERATURE REVIEW)*

Literatür çalışması kapsamında; yükseköğretim kurumlarında dijitalleşme süreçleri ve öğrenci destek hizmetleri ile ilgili çalışmalar incelenmiştir. Geleneksel öğretim gören öğrencilerin bizzat başvurduğu öğrenci işlerinin hizmet kalitesine yönelik yapılan bir çalışmada değişik bölümlerden seçilen öğrencilere anket çalışması uygulanmış ve sonuç olarak öğrenci talebi üzerine hazırlanan belgelerin zamanında hazırlanmaması ve not girişlerinde yaşanan sorunlar en çok şikâyet edilen konular olurken hizmet kalitesi algısı sıralamasında ne önemli olan unsur güvenilirlik olarak belirlenmiştir [5]. Genç Kumtepe ve diğerleri [6] yükseköğretim kurumlarında öğrenme topluluklarının oluşması ve devamı için

destek hizmetleri sağlanması gerektiğini vurgulamış ve bu uğurda iyi yapılanmış öğrenme merkezleri, bölgesel ağlar ve bilgi iletişim teknolojilerinin sürece dahil olması gerektiğini belirtmiştir.

Yüksek öğrenimde dijitalleşmeyi öğrenci perspektifinden değerlendiren Thoring ve diğerleri [7] öğrencilerin dijitalleşme ile ilgili ilk beklentilerinin ders notlarının ve üniversite bireyleri ile çevrimiçi etkileşim olanaklarının dijital olarak sağlanması olduğunu vurgulamışlardır. Öğrencilerin Google, Microsoft vb. gibi büyük sağlayıcılardan etkilenen kullanıcı alışkanlıkları nedeniyle beklentilerinin yüksek olduğunu da çalışmalarında belirtmişlerdir. Bununla birlikte öğrenciler, üniversitenin bilgi işlem hizmetlerinde küçük iyileştirmeler şeklinde yapılan dijitalleşme derecesini güçlendiren değişikliklerden memnun olacaklarını belirtmişlerdir.

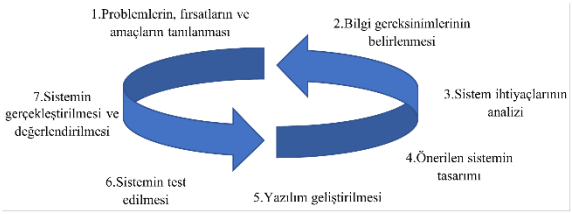
Yükseköğretim kurumlarında dijital dönüşüme yönelik çalışmaları inceleyen Taşçı ve Taşlıbeyaz [1] 2015 ve 2019 yılları arasında Türkiye ve diğer ülkelerde bu konu ile ilgili yapılan yayınları araştırmıştır. Çalışmaların tarandığı veritabanları Web of Science, EBSCO, ERIC ve Google Akademik olarak belirtilmiştir. Araştırma kriterlerine uygun olan 22 çalışma incelenmiş ve sonuç olarak son yıllarda bu alanda yapılan çalışmaların arttığı ve nitel araştırma yöntemlerinin daha fazla kullanıldığını belirtmiştir.

Aggarwal ve Verma [8] modern web uygulamalarının geliştirilmesinde kullanılan en popüler yığınlar olan MEAN ve MERN yığınları için seçim sırasında önemli rol oynayan temel faktörleri değerlendiren bir çalışma gerçekleştirmişlerdir. Her iki yığında arka-uç geliştirmede Node.js ve Express.js kullanılırken veritabanı olarak MongoDB tercih edilmektedir. İki yığının birbirinden farkı ise MEAN yığınında ön-uç geliştirme için Angular tercih edilirken MERN yığınında React.js kullanılmasıdır. Her iki yığının da JavaScript altında toplanması, geliştiricilerin hem ön-uç hem de arka-uç işlemlerinde farklı dilleri anlama ihtiyacı duymaması, geliştirme maliyetlerinde büyük bir düşüşe yol açmakta ve verimliliği artırmaktadır. Çalışmanın sonucu olarak her iki yığının da hızlı ön uç geliştirme için son derece güvenilir çerçeveler olduğu belirtilmiştir. En büyük farkın ise her iki yığının yapılandırılma ve inşa edilme biçiminde olduğu vurgulanmıştır. MEAN yığını büyük ölçekli uygulamalar için daha iyi bir seçenek olarak görülürken, MERN yığını daha küçük uygulamaların daha hızlı geliştirilmesinde tercih edilmektedir.

Daşdemir ve Kara [9] çalışmalarında NoSQL veritabanları arasında seçim yapabilmek için kriter olarak değerlendirilen tepki verme sürelerini incelemişlerdir. Çalışma içerisinde popüler NoSQL çözümler olarak görülen MongoDB, CassandraDB ve OrientDB veritabanları karşılaştırılmıştır. Yahoo bulut hizmet kıyaslama aracından elde edilen 6 farklı iş yükü kullanarak gerçekleştirilen testlerde gerçekleştirilmesi istenen hedefi yakalamada en iyi sonucu MongoDB verirken, gecikme süresi sonuçlarında en başarılı OrientDB olmuştur. CassandraDB neredeyse bütün testlerde diğer iki veritabanından daha kötü sonuçlar elde etmiştir.
Bilgi teknolojileri kullanılarak gerçekleştirilen öğrenci destek hizmetleri yönünden incelenen çalışmaların neredeyse tamamının açık ve uzaktan eğitime yönelik hizmet ve destek süreçlerine yönelik olduğu görülmüştür. Geleneksel eğitim-öğretim şartları göz önüne alınarak, fiziksel kampüste aktif olarak öğretim gören öğrencileri kapsayan nitelikte olması ile bu çalışma benzerlerinden ayrılmaktadır.

3. YÖNTEM *(METHOD)*

Çalışma kapsamında belirtilen problemlere çözüm olması, öğrenci işleri birimi ve öğrenciler arasında destek iletişimini sağlaması adına geliştirilen yazılım sistem geliştirme yaşam döngüsü adımları takip edilerek geliştirilmiştir (Şekil 1).



Şekil 1. Sistem geliştirme yaşam döngüsü [10]
(System development life cycle)

Problemlerin, fırsatların ve amaçların tanımlanması adımında DEÜ-İİBF bünyesinde bulunan öğrenci işleri biriminde öğrencilerle iletişim kurmak için herhangi bir yazılım kullanılmaması ve öğrenci taleplerinin fiziksel başvuru ile yapılmasından kaynaklanan durumlar çalışmanın problemi olarak tanımlanmıştır. Öğrenci-personel iletişimini güçlendirmek, öğrenci desteği kaynaklı iş yükünü görüntülenebilir kılmak ve iş takibi yapabilmek adına öğrenci destek platformu geliştirmesi çalışmanın amacı olarak tanımlanmıştır. Bilgi gereksinimlerinin belirlenmesi adına DEÜ-İİBF öğrenci işleri birimi ile odak görüşme gerçekleştirilmiş, yaşanılan problemler tartışılmış ve birimin geliştirilen yazılım ile ilgili beklentileri araştırılmıştır.

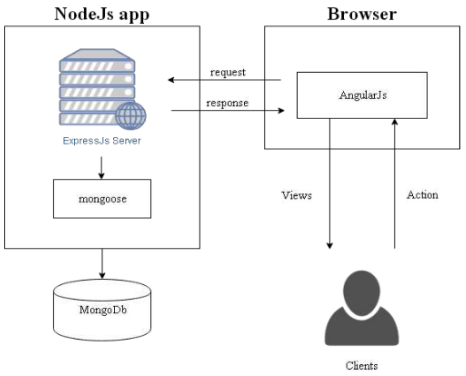
3.1. MEAN Yığını *(MEAN Stack)*

Yazılımın geliştirilmesi aşamasında MEAN yığını (Stack) yeteneklerinden yararlanılmıştır. MEAN stack dinamik web ve mobil uygulama geliştirme sürecini

hızlandırmaya yardımcı olan JavaScript (JS) tabanlı bir yapıdır. MEAN içerisindeki her bir bileşen JavaScript Nesnesi Gösterimi (JavaScript Object Notation-JSON) dilini kullandığı için ara yüz, sunucu ve veritabanı arasında gerçekleştirilen işlemleri birbiri ile uyumlu bir hale getirmeye gerek olmaması yığının avantajlı taraflarından biridir [11]. Web uygulamalarında Üst metin İşaretleme Dili (Hyper Text Markup Language-HTML) ve Basamaklı Stil Sayfaları (Cascade Style Sheets-CSS) web arayüzünün sunumu ile ilgilenirken JavaScript sayfanın interaktif çalışmasını sağlayan fonksiyonlarını belirlemektedir. MEAN yığınının baş harfleri bu yapının içerdiği bileşenleri işaret etmektedir.

M, veritabanı olarak kullanılan MongoDB'yi; **E,** sunucu tarafında kod ve mantık yazmayı basitleştiren bir Node.js çerçevesi olan Express.js'yi; **A,** tek sayfa uygulamaların (Single Page Asslications-SPAs) oluşturulmasına izin veren bir istemci tarafı (tarayıcı) çerçevesi olan Angular'ı; **N,** başlarda sadece tarayıcı içerisinde çalışan JavaScript'in sunucu tarafında çalıştırılmasını sağlayan kütüphane olan Node.js'yi temsil etmektedir.

MEAN stack "Tam yığın" (Full stack) geliştirmenin popüler bir örneğidir. Full stack kavramı geleneksel tanıma göre; bir web uygulamasını geliştiren kişinin hem ön-uç (frontend) hem de arka-uç (backend) ile ilgili çalışmasıdır. Full stack geliştirme yapan birinin web uygulamasının sadece kullanıcıya yönelik özelliklerini değil, bu uygulamanın içeriğini depolaması ve iletmesi için kullanılan veritabanlarıyla beraber tüm sunucu işlemlerini gerçekleştirmesi beklenir [12]. MEAN stack yapısının çalışma şeklini ifade eden görsel Şekil 2'de sunulmuştur.



Şekil 2. MEAN yığını yapısı [13]
(MEAN stack structure)

MEAN yığını kullanılarak geliştirilen sistem için belirlenen gereksinimler şu şekildedir:

- Platformlar arası çalışan MEAN yığını geliştirmesi için Windows, macOS ya da Linux işletim sistemlerinden biri tercih edilebilir.
- Visual Studio Code, Atom, Sublime Text gibi metin düzenleyiciler ya da bütünleşik geliştirme ortamları (Integrated Development Environment- IDE) kodlama için gereklidir.
- Node.js, JavaScript'i sunucu tarafında çalıştırmak için gerekli olan bir JavaScript çalıştırma ortamıdır.
- Angular, dinamik web uygulamaları oluşturmak için kullanılan bir ön-uç JavaScript çerçevesidir.
- Express.js, web uygulamaları ve uygulama programlama arayüzleri (Asslication Programming Interface-API) oluşturmak için kullanılan bir arka-uç JavaScript çerçevesidir.
- MongoDB veritabanı için masaüstü uygulamasının kullanılması durumunda son sürümünün yüklenmiş olması gereklidir. Web arayüzü kullanılacaksa MongoDB'nin kendi internet sitesinde yer alan "Atlas" isimli veritabanı oluşturma ve düzenleme aracı için kullanıcı hesabı oluşturulmalıdır.
- Node.js, Angular ve Express.js'nin en son sürümlerinin geliştirme yapılacak olan bilgisayara kurulumlarının gerçekleştirilmesi gereklidir.
- Ek kütüphaneler olarak MEAN yığın geliştirme, veritabanına bağlantı için Mongoose, Body-parser, Nodemon ve Nodemailer gibi ek kitaplıklar ve bağımlılıklar gerektirir.
- Sorunsuz bir geliştirme için iyi bir internet bağlantısına ve minimum 4 GB RAM'e sahip olmak gerekmektedir.

Geliştirilen sistemin teknik yapısı ön-uç (frontend), arka-uç (backend) ve veritabanı (database) olarak 3 ana başlığa ayrılmıştır. Yazılımın arka plan işlemlerini içeren frontend ve kullanıcı arayüzlerinde yapılan işlemleri içeren backend olarak iki farklı kısımda geliştirilmesi sistemin çalışma performansını olumlu etkilediği için tercih edilmiştir.

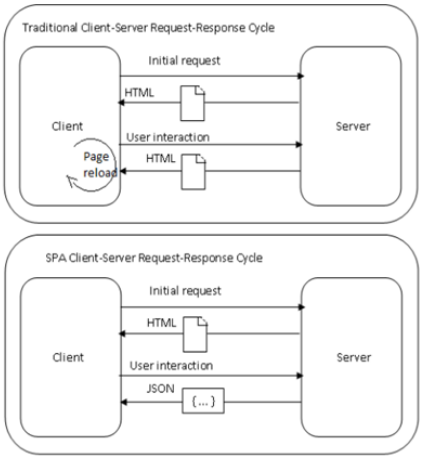
3.1.1. Ön-uç *(Frontend)*

Sistemin kullanıcı arayüzlerinde çalışan kısmıdır. Ön-uç geliştirme, modern web uygulamaları oluşturmanın önemli bir yönüdür. Kullanıcı arayüzü oluşturmayı ve sorunsuz bir kullanıcı deneyimi sağlamayı içerir. Bu çalışma kapsamında geliştirilen sistemin frontend kısmında yazılımın tasarımı, kodlama ve elementlerin ekranda yerleşimi için temel olarak HTML ve CSS kodlama dilleri tercih edilmiştir. Son kullanıcının ekranları daha efektif kullanabilmesi ve yükü sunuculardan ziyade istemci (client) tarayıcılarına dağıtmak için JavaScript tabanlı MEAN yığının bir bileşeni olan Angular kullanılmıştır. Popüler bir JavaScript çerçevesi olan Angular, frontend geliştirme için güçlü bir araçtır. Angular ile geliştiriciler, birden çok platform ve cihazda sorunsuz çalışan dinamik ve

etkileşimli web uygulamaları oluşturabilir [14]. Genel olarak, web uygulamalarında frontend için Angular kullanmak, sistemin işlevselliğini, performansını ve kullanıcı deneyimini büyük ölçüde artırabilir.

Tek sayfa uygulamalar (Single Page Asslications-SPAs) internetin ilk zamanlarında geliştirilen geleneksel web sitelerinden farklı olarak tüm sayfayı yenilemeye gerek kalmadan bağımsız olarak değiştirilebilen ayrı bileşenlerden oluşur, böylece her kullanıcı eyleminde tüm sayfanın yeniden yüklenmesi gerekmez [15]. Birden çok sayfa arasında gezinmek yerine, kullanıcı aynı sayfada kalır ve içerik gerçek zamanlı olarak güncellenir.

Angular, geliştiricilere zengin, dinamik ve etkileşimli SPAs oluşturmak için ihtiyaç duydukları araçları sağlar. İki yönlü veri bağlama, bağımlılık ekleme ve bileşen tabanlı mimari gibi güçlü özellikleri, büyük miktarda veriyi ve kullanıcı etkileşimlerini işleyebilen karmaşık uygulamalar oluşturmayı kolaylaştırır. İki yönlü veri bağlama, görünümdeki verilerin değiştirilmesinin modeldeki verileri güncellediği anlamına gelir. Aynı şekilde, modelde değiştirilen veriler de görünüme yansıtılır [16]. Angular'ın SPAs oluşturmaya odaklanması, ekran boyutuna duyarlı (responsive) ve yüksek düzeyde etkileşimli web uygulamaları oluşturmak isteyen geliştiriciler için popüler olarak tercih edilme nedeni hale gelmiştir. İstemci-sunucu istek-yanıt döngüsünün geleneksel web sayfası ve SPAs açısından karşılaştırılması Şekil 3'teki gibidir.



Şekil 3. İstemci-sunucu yanıt döngüsü [15] *(Client-Server)*

3.1.2. Arka-uç *(Backend)*

Sistemin sunucu tarafında çalışan arka plan işlemleri şeklinde belirtilen bölümüdür. Backend geliştirme veritabanlarıyla etkileşim kuran, kullanıcı kimlik doğrulaması yapabilen ve frontend ile iletişim kuran uygulamanın sunucu tarafını oluşturmayı içerir. Backend bölümünde bu çalışma kapsamında sistem geliştirme için kullanılan MEAN stack yapısı içinde bulunan Node.js ve bu yapı üzerinde çalışan Express.js kullanılmıştır. Node.js, ilk zamanlarında sadece tarayıcı içerisinde çalışması için tasarlanan JavaScript'i sunucu tarafında da çalıştırmayı sağlayan farklı işletim sistemlerinde kullanılabilen bir çalışma ortamıdır (run-time environment) [17].

Express.js, web uygulamalarının geliştirilmesinde esnekliğe ve yüksek düzeyde özelleştirmeye imkân sağlayan, Node.js üzerinde çalışan bir web çerçevesidir [18]. Üst metin Transfer Protokolü (Hypertext Transfer Protocol-http) isteklerini ve yanıtlarını ele almak, yönlendirmeleri uygulamak ve orta yazılımları (middleware) ele almak gibi işlemleri basitleştirir ve geliştiricilerin güçlü ve ölçeklenebilir sunucu uygulamaları oluşturmasını kolaylaştırır.

API, "Asslication Programming Interface" kelimelerinin kısaltmasıdır ve uygulamaların kullanıcı etkileşimi olmadan bir ağ üzerinden birbirleriyle konuşmaları için çerçeve tanımlayan yazılımdan yazılıma bir arayüzdür [19]. Bir API, uygulamanın belirli işlevlerini diğer uygulamalar tarafından kullanılabilir hale getirir. API'ler, yazılım geliştiricilerin işlerini kolaylaştırır, zaman ve maliyet tasarrufu sağlar ve farklı platformlar arasında entegrasyonu kolaylaştırır. Bir API, önceden tanımlanmış bir dizi komut, protokol ve arayüzden oluşur ve bu sayede bir uygulamanın diğer uygulamalarla iletişim kurabilmesi ve veri paylaşımı yapabilmesi sağlanır. Örneğin, bir web uygulaması, bir API kullanarak bir harita hizmetinden konum verileri alabilir veya bir ödeme sistemi API'si kullanarak ödeme işlemlerini gerçekleştirebilir.

3.1.3. Veritabanı *(Database)*

Günümüzde sürekli gelişen ve değişen teknolojik özellikler nedeni ile farklı çeşitlerde veritabanları bulunmaktadır ve yazılım geliştirme konusunda en önemli adımlardan biri veritabanı seçimidir. Yapılan bir çalışmaya göre bu seçimde en önemli kriterler sırasıyla güvenlik, yazılım dilleri ile uygunluk ve kullanım kolaylığı olarak belirtilmiştir [20].

Bu makale içerisinde geliştirilen sistemde MEAN yığınının veritabanına işaret eden bileşeni olan doküman tabanlı olarak çalışan MongoDB tercih edilmiştir. MongoDB ilişkisel olmayan NoSQL (Not Only Structured Query Language) bir veritabanıdır. SQL (Structured Query Language) veritabanı ile temel farkı kayıtları tablolarda tutmak yerine

koleksiyon olarak dokümanlar içerisinde tutmaktadır. Doküman odaklı veritabanı sistemleri, JSON gibi formatların yaygınlığı nedeniyle geliştirme ve dağıtım hızının kritik olduğu modern web uygulamaları geliştirme alanında popüler olmuştur [21].

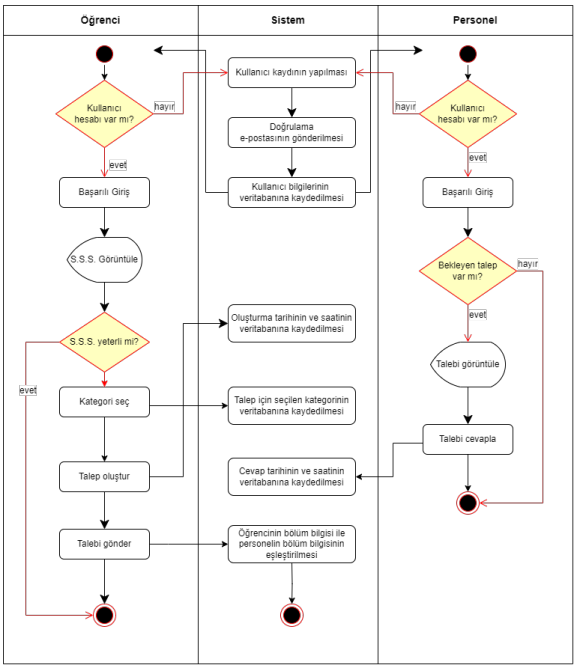
MongoDB, ilişkisel veritabanı sistemlerinden farklı olarak, yapısal bir şemaya sahip olmak zorunda değildir. Bu, verilerin daha esnek bir şekilde saklanabilmesi anlamına gelir. MongoDB özellikle bir yığın olarak kullanılan MEAN stack gibi hızlı geliştirme ortamlarında kullanışlı olduğu için tercih edilmektedir. MongoDB içerisinde JSON formatında tutulan veriler backend tarafında yazılan bir metot ile JSON olarak alınıp doğrudan ön-uca (frontend) yine JSON olarak gönderilebildiği için tercih edilen bir veritabanıdır. Bir MongoDB veritabanı, bir veya daha fazla koleksiyon içerebilir. Koleksiyonlar, belgelere benzeyen verileri içerir ve her bir belge, alanlar ve değerlerle tanımlanır. Örneğin, bir kullanıcı koleksiyonu, her bir kullanıcının adı, e-posta adresi, şifresi ve diğer bilgilerini içerebilir ve her kullanıcıya farklı bilgiler de tanımlanabilir.

MongoDB veritabanının yönetim işlemleri için web tabanlı MongoDB Atlas uygulaması kullanılmaktadır. MongoDB Atlas, MongoDB'yi oluşturan kişiler tarafından geliştirilen bir çoklu bulut veritabanı hizmetidir. Seçilen bulut sağlayıcılarında (Amazon Web Service, Microsoft Azure ya da Google Cloud Platform) esnek ve performanslı küresel uygulamalar oluşturmak için ihtiyaç duyulan çok yönlülüğü sunarken veritabanını dağıtmayı ve yönetmeyi basitleştirir [22]. Atlas, bulutta kurulumu, işletimi ve ölçeklendirmesi kolay olan MongoDB tarafından sunulan servis tabanlı doküman deposu (Document Store as a Service-DSaaS) kavramının bir örneğidir [21].

4. YAZILIMIN GELİŞTİRİLMESİ *(SOFTWARE DEVELOPMENT)*

Çalışma kapsamında geliştirilen sistemin işleyiş şeklini gösteren bir akış diyagramı hazırlanmıştır. Diyagram, analiz ve tasarım için ortak ve standart bir yapı oluşturmak amacıyla ortaya konulan bir notasyon olan Birleşik Modelleme Dili (Unified Modelling Language-UML) kullanılarak oluşturulmuştur [23].

Şekil 4'te sunulan diyagram, sistem içerisinde kullanıcı hesabı bulunan 2 farklı tip kullanıcının gerçekleştirdiği işlemleri göstermektedir. Bunlardan biri mevcutta öğretim gören öğrencilerden oluşurken diğeri öğrenci işleri birimi personelinden oluşmaktadır. İlk adım olarak öğrenciler için sistemde kullanıcı hesabı oluşturma adımı bulunmaktadır. Bu aşamada öğrenciden bilgileri istenerek belirtmiş olduğu e-posta adresine doğrulama e-postası gönderilmektedir. E-postasını doğrulamayan öğrenci sistem içerisinde işlem yapamamaktadır. Bu işlem ile öğrencinin aktif olarak kullandığı güncel e-postasının elde edilmesi amaçlanmaktadır. Öğrenci sisteme giriş yaptığında sıkça sorulan soruları kategorilere ayrılmış bir şekilde görüntülemektedir. Bu sorular arasında danışmak istediği konu ile ilgili içerik bulamazsa talep oluşturma işlemi ile danışmak istediği konuyu öğrenci işleri birimi personeline sistem içerisinden iletmektedir. Öğrenci işleri biriminde görev dağılımı fakültedeki bölümlere göre yapılmaktadır; her personel bir ya da birden fazla bölümden sorumlu olup, bu bölümde/bölümlerde öğretim gören öğrencilerin işlemlerini yerine getirmektedir. Bu durum göz önüne alınarak sistem içerisinde öğrenci tarafından açılan talep esnasında öğrencinin bölüm bilgisi tutulmakta ve talep o bölümden sorumlu olan personele gönderilmektedir.



Şekil 4. Sistem akış şeması
(System flowchart)

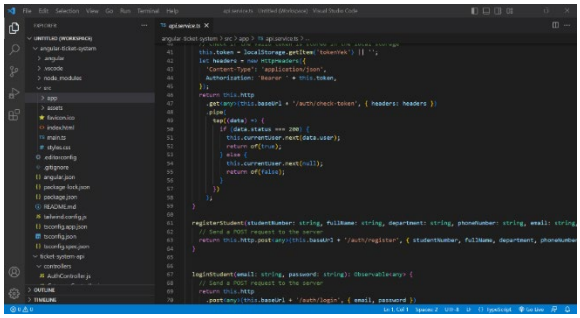
Öğrenci talep oluşturma aşamasında, ilk olarak oluşturacağı talebin hangi kategoride olduğunu belirtir. Kategori seçim işlemi ile sistem içerisinde oluşturulan talepler için sınıflandırma yapılması ve gelecek çalışmalarda bu sınıflandırma kullanılarak istatistiki veriler elde edilmesi amaçlanmaktadır.

Öğrenci işleri birimi personeli sisteme giriş yaptığında kendisine gönderilen bir talep olup olmadığını görüntülemektedir. Bekleyen bir talep olması durumunda personel talebi görüntüler ve uygun bir şekilde cevaplar. Personel tarafından eklenen cevap, talebi oluşturan öğrenciye gönderilir. Sistem

içerisinde işlem gören taleplerin öğrenci tarafından oluşturulması ve personel tarafından cevaplanmasına yönelik tarih ve saat verileri veritabanına kaydedilmektedir. Bu veriler ile gelecek çalışmalarda ortalama cevap süresi vb. istatistiki veriler elde edilmesi amaçlanmaktadır.

4.1. Geliştirme Ortamı *(Development Environment)*

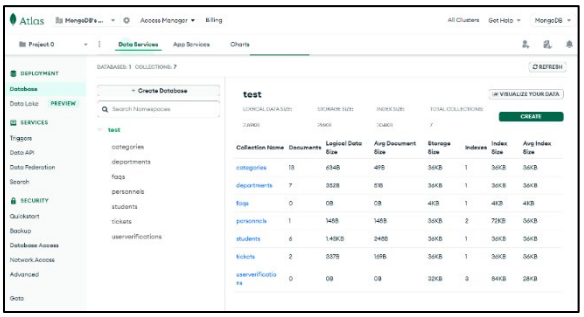
Geliştirilen sistemin kodlama aşamasında masaüstü çalışan bir kaynak kodu düzenleyicisi olan Microsoft Visual Studio Code (VS Code) kullanılmıştır (Şekil 5). Çalışma kapsamında VS Code'un öne çıkan tercih edilme nedeni JavaScript, TypeScript ve Node.js için yerleşik destek ile birlikte sunulmasıdır [24]. Bununla beraber VS Code içerisinde yer alan MongoDB eklentisi ile sağlanan veritabanı bağlantısı çalışma kapsamında kullanılan MongoDB veritabanının yönetim işlemlerinin VS Code içerisinden gerçekleştirilmesini mümkün kılmaktadır.



Şekil 5. VS Code ekran görüntüsü
(VS Code screenshot)

4.2. MongoDB Veritabanı *(MongoDB Database)*

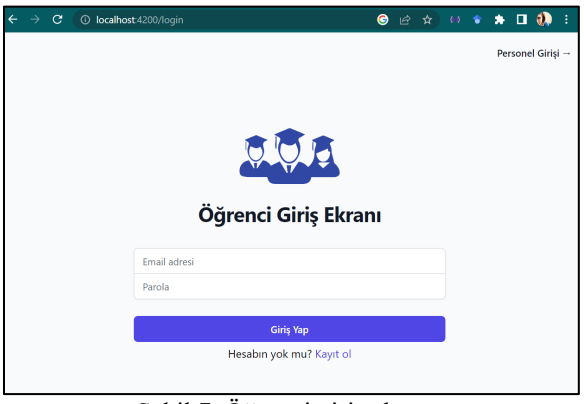
Veritabanı olarak; çalışmaya konu olan sistemin geliştirilmesi aşamasında MEAN yığının bir bileşeni olan MongoDB veritabanı kullanılmıştır. NoSQL veritabanı olan MongoDB veritabanın oluşturulması ve yönetimi aşamasında web arayüzünde bulunan Atlas isimli veritabanı yönetim aracı kullanılmıştır (Şekil 6). Atlas içerisinden sistemde işlem görecek olan veritabanı oluşturulmuş ve kod bağlantısı gerçekleştirilmiştir. Bunun için backend tarafında işlem gören Node.js modülü içerisinde bir Nesne Veri Modelleme (Object Data Modeling-ODM) kütüphanesi olan Mongoose kullanılmıştır. Mongoose, MongoDB ve Node.js arasında gerekli veri yapısını sağlayan bir bağlantı görevi görür ve veri depolama ve almada esnekliği korur [11].



Şekil 6. MongoDB Atlas veritabanı yönetim aracı
(MongoDB Atlas database management tool)

4.3. Öğrenci Giriş Ekranı *(Student Login Interface)*

Çalışma kapsamında MEAN yığını kullanılarak geliştirilen destek yönetim sistem içerisinde Dokuz Eylül Üniversitesi İktisadi ve İdari Bilimler Fakültesinde öğrenim gören öğrencilerin sisteme giriş yapabildiği arayüz Şekil 7'deki gibidir. Sistemde daha önce kullanıcı hesabı oluşturmuş olan öğrenci bu bölümde e-posta adresi ve şifre bilgisi ile sisteme giriş yapabilmektedir.

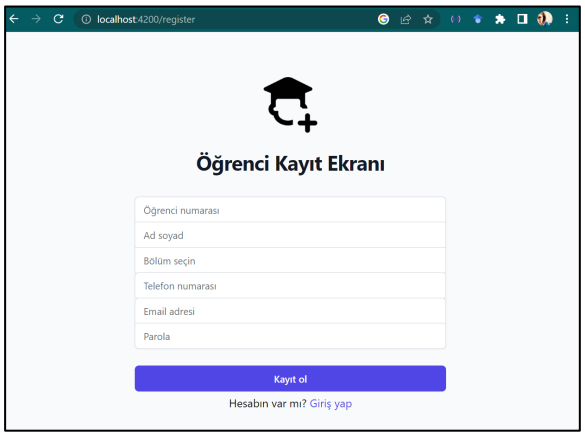


Şekil 7. Öğrenci giriş ekranı
(Student login interface)

Sistemin backend tarafında işlem gören Node.js modülünün bir parçası olan postLogin işlevi, kullanıcılardan gelen oturum açma isteklerini işlemektedir. Önce, giriş ekranında belirtilen e-posta adresine sahip bir kullanıcının öğrenci koleksiyonunda (MongoDB veritabanında öğrenci bilgilerinin yer aldığı koleksiyon) olup olmadığını kontrol eder. Değilse, bir hata yanıtı döndürür. Belirtilen bilgilerle eşleşen bir kullanıcı bulunursa, kullanıcının e-posta adresini doğrulayıp doğrulamadığını kontrol eder. Kullanıcı doğrulandıysa, sağlanan parolanın veritabanında saklanan şifreli parola ile eşleşip eşleşmediğini kontrol eder. Değilse, bir hata yanıtı döndürür. Parola eşleşirse, kullanıcının bilgilerini içeren bir JWT (JSON Web Tokens) belirteci oluşturur ve belirteçle birlikte bir JSON yanıtı döndürür.

4.3.1. Kullanıcı hesabının oluşturulması *(Registration of a user)*

Öğrencinin sistem içerisinde kullanıcı kaydının bulunmaması durumunda yeni bir kullanıcı hesabı oluşturması gerekmektedir. Kullanıcı kaydının oluşturulması için geliştirilen arayüz Şekil 8'de gösterilmiştir. Sistemin backend tarafında Node.js modülünde işlem gören postRegister işlevi, kullanıcılardan gelen kayıt isteklerini işler. İlk önce aynı öğrenci numarasına veya e-posta adresine sahip bir kullanıcının veritabanında var olup olmadığını kontrol eder. Değilse, sağlanan bilgilerle öğrenci koleksiyonunda (MongoDB veritabanında öğrenci bilgilerinin yer aldığı koleksiyon) yeni bir kullanıcı oluşturur ve kullanıcıların parolalarını güvenli bir şekilde saklamasını sağlayan bir parola kriptolama algoritması olan bcrypt kullanarak parolayı şifreler.



Şekil 8. Kullanıcı hesabının oluşturulması *(Registration)*

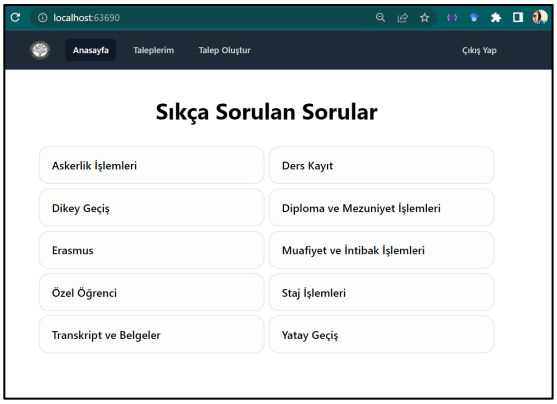
4.3.2. Kullanıcı hesabının doğrulanması *(Verification of a user)*

Kullanıcı hesabının oluşturulması aşamasında öğrencinin belirttiği e-posta adresine sistem tarafından doğrulama linki gönderilir. Öğrencinin sistem içerisinde işlem yapabilmesi için bu adımı tamamlaması gerekmektedir. E-posta doğrulama işlemi, backend tarafında işlem gören kimlik doğrulama ve kaydı işleyen bir Node.js modülüdür. Modül, HTTP isteklerini işlemek için çeşitli işlevleri dışa aktarır. Kullanıcı hesabı oluşturma aşamasında çalışan postRegister işlevi bir doğrulama kodu oluşturur, UserVerification koleksiyonuna (MongoDB veritabanında doğrulama kodlarının tutulduğu koleksiyon) kaydeder ve kullanıcıya hesabını doğrulamak için bağlantı içeren bir e-posta gönderir. İşlev, kayıt başarılı olursa bir başarı mesajı veya bir hata oluşursa bir hata mesajı içeren bir JSON yanıtı döndürür.

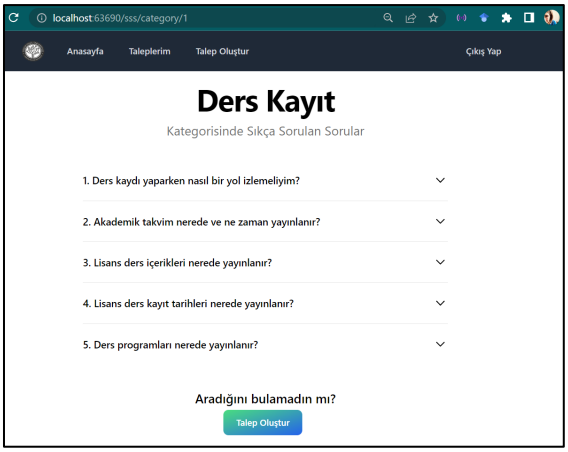
Yine arka-uç tarafında işlem gören Node.js modülü içerisindeki VerifyEmail işlevi, kullanıcılardan gelen e-posta doğrulama isteklerini işler. Önce, öğrenci koleksiyonunda sağlanan kimliğe sahip bir kullanıcının olup olmadığını kontrol eder. Değilse, bir hata yanıtı döndürür. Bir kullanıcı bulunursa, UserVerification koleksiyonunda sağlanan kimlik ve doğrulama koduyla eşleşen bir doğrulama kodu olup olmadığını kontrol eder. Değilse, bir hata yanıtı döndürür. Bir doğrulama kodu bulunursa, kullanıcının isVerified alanını true olarak günceller ve doğrulama kodunu UserVerification koleksiyonundan siler. İşlev, e-posta doğrulanırsa bir başarı mesajı veya bir hata oluşursa bir hata mesajı içeren bir JSON yanıtı döndürür.

4.3.3. Anasayfa *(Homepage)*

Kullanıcı hesabı oluşturan öğrenci sisteme başarılı bir şekilde giriş yaptıktan sonra anasayfada DEÜ-İİBF bünyesindeki eğitim öğretim süreçlerini kapsayan sıkça sorulan soruları kategorize edilmiş bir şekilde görüntüler (Şekil 9).



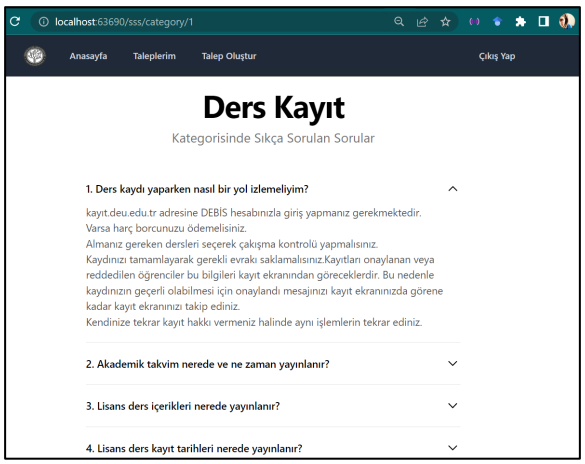
Şekil 9. Anasayfa *(Homepage)*



Şekil 10. Sıkça sorulan sorular-1 *(Frequently asked questions-1)*

Seçilen kategori altında bu konuda yer alan sorular listelenmektedir (Şekil 10 ve Şekil 11). Öğrenci kullanıcısı için anasayfada yer alan yönlendirme

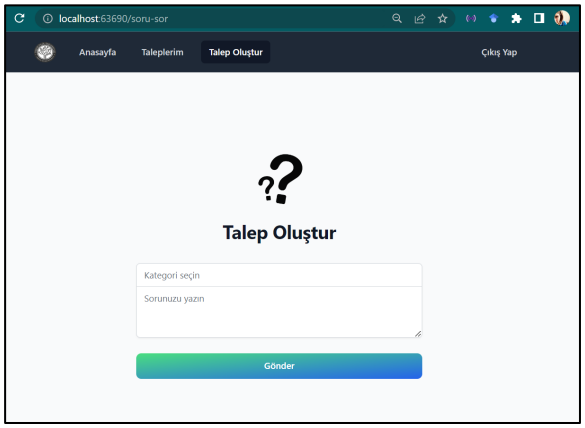
çubuğunda (navigasyon bar) daha önce açılmış olan talepler ve talep oluşturma işlemlerine erişim sağlanmaktadır.



Şekil 11. Sıkça sorulan sorular-2
(Frequently asked questions-2)

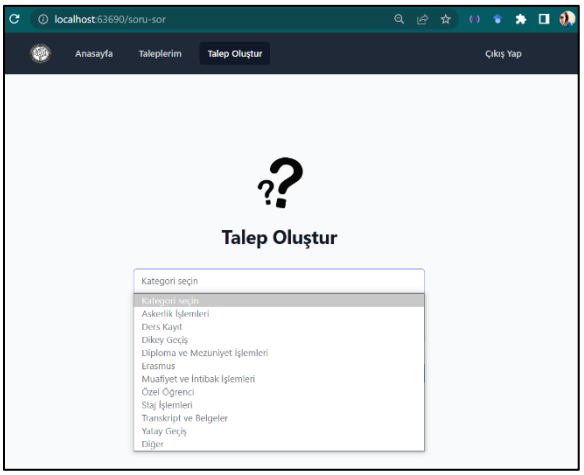
4.3.4. Talep oluşturma işlemi *(Creation of a request)*

Sıkça sorulan sorular içerisinde danışmak istediği konuyu ya da soruyu bulamayan öğrenci için sistem içerisinde talep oluşturma işlemi bulunmaktadır (Şekil 12). Öğrenci bu ekrandan danışmak istediği konu ile alakalı bir kategori seçtikten sonra talebini yazılı bir şekilde belirtmektedir (Şekil 13).



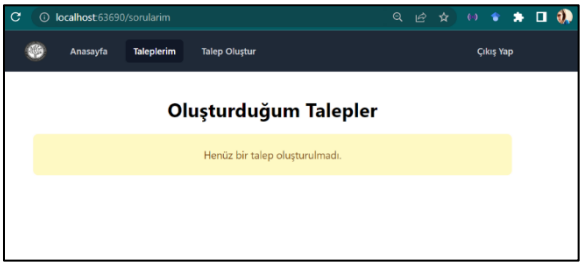
Şekil 12. Talep oluşturma ekranı
(Interface of creation of a request)

DEÜ-İİBF öğrenci işleri biriminde görev bölümü fakültedeki bölümlere göre yapılmaktadır. Bir personel bir ya da birden fazla bölümden sorumlu olup, bu bölüm ya da bölümlerde öğretim gören öğrencilerle ilgili işlemleri gerçekleştirmektedir. Geliştirilen sistemin veritabanı yapısı bu duruma uygun olup hem personel hem de öğrenci koleksiyonunda bölüm bilgisi tutulmaktadır. Talep oluşturma aşamasında talebi oluşturan öğrenci ile cevaplayan personel arasında eşleştirme yapılmaktadır. Öğrenci tarafından oluşturulup gönderilen talep, öğrencinin veritabanında kayıtlı olan bölüm bilgisine bakılarak, bu bölüm bilgisi ile eşleşen personele gönderilmektedir.

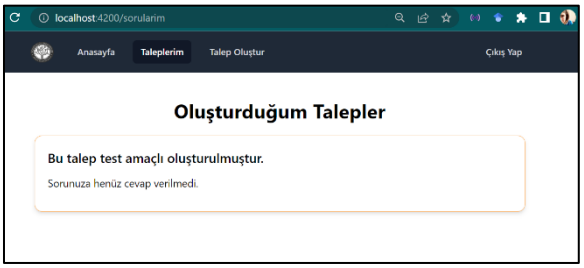


Şekil 13. Talebe ait kategori seçimi
(Category selection of the request)

Sistem içerisinde öğrenci tarafından daha önce oluşturulan talepler "Taleplerim" ekranında listelenmektedir (Şekil 14 ve Şekil 15).



Şekil 14. Oluşturulan talepler-1
(Created requests-1)

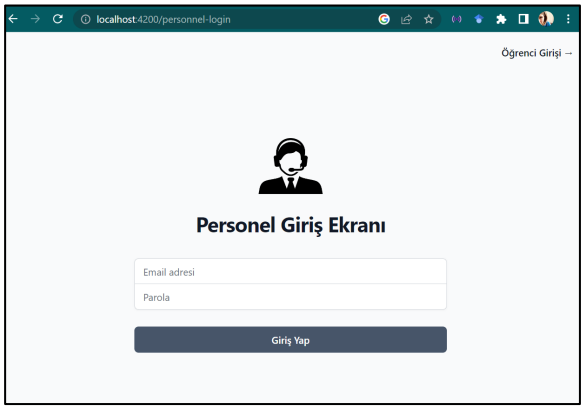


Şekil 15. Oluşturulan talepler-2
(Created requests-2)

4.4. Personel Giriş Ekranı *(Staff Login Interface)*

Çalışma kapsamında MEAN yığını kullanılarak geliştirilen destek yönetim sistem içerisinde DEÜ-İİBF öğrenci işleri biriminde görevli personelin sisteme giriş yapabildiği arayüz Şekil 16'da gösterilmektedir. Sistemde daha önce kullanıcı hesabı

oluşturmuş olan personel bu bölümde e-posta adresi ve şifre bilgisi ile sisteme giriş yapabilmektedir.

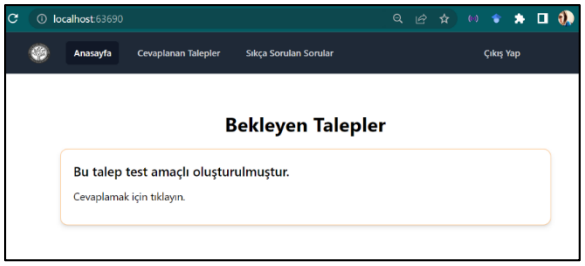


Şekil 16. Personel giriş ekranı
(Staff login interface)

Sistemin backend tarafında işlem gören Node.js modülünün bir parçası olan PersonnelLogin işlevi, personelden gelen oturum açma isteklerini işler. Önce, sağlanan e-posta adresine sahip bir personelin Personel koleksiyonunda (MongoDB veritabanında personel bilgilerinin yer aldığı koleksiyon) olup olmadığını kontrol eder. Değilse, bir hata yanıtı döndürür. Eğer bir personel bulunursa, sağlanan parolanın veritabanında şifreli olarak saklanan parola ile eşleşip eşleşmediğini kontrol eder. Değilse, bir hata yanıtı döndürür. Parola eşleşirse, personelin bilgilerini içeren bir JWT (JSON Web Tokens) belirteci oluşturur ve belirteçle birlikte bir JSON yanıtı döndürür.

4.4.1. Anasayfa *(Homepage)*

Sisteme başarılı bir şekilde giriş yapan personel kullanıcısı anasayfada öğrenci/öğrenciler tarafından (varsa) açılmış olan talepleri görüntüler (Şekil 17).

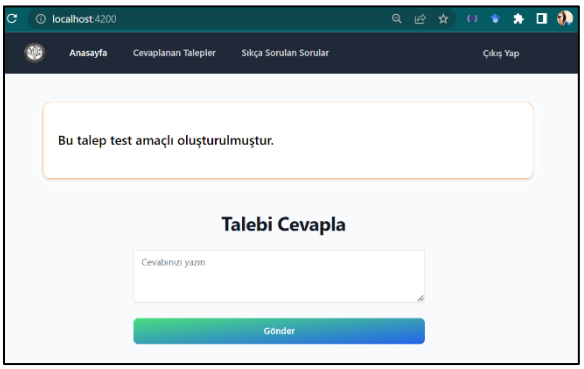


Şekil 17. Anasayfa
(Homepage)

Personel kullanıcısı için anasayfada yer alan yönlendirme çubuğunda (navigasyon bar) daha önce cevaplanmış olan talepler ve öğrenciler için oluşturulmuş olan sıkça sorulan sorular ekranına erişim sağlanmaktadır.

4.4.2. Talep Cevaplama İşlemi *(Request Answering Process)*

DEÜ-İBBF öğrenci işleri biriminde görevli personel kullanıcısı, fakülte içerisindeki sorumlu olduğu bölümün öğrencisinden gelen talebe Şekil 18'deki arayüzden cevap verebilmektedir. Cevaplanan talep talebi oluşturan öğrencinin kullanıcı hesabına gönderilir.



Şekil 18. Talep cevaplama ekranı
(Interface of requests answering)

5. SONUÇ VE TARTIŞMA *(CONCLUSION AND DISCUSSION)*

İçinde bulunduğumuz dijital çağda hemen her bireyin internet erişimi sağlayabilmesi ile kullanılmakta olan teknolojik araçların hızlı gelişimi kaçınılmaz olmuştur. Bununla beraber zaman içerisinde kurum ve kuruluşlardaki fiziksel süreçlerin dijital ortama aktarılmasını içeren dijitalleşme çalışmaları Yükseköğretim kurumlarını da etkilemiştir. Yükseköğretimde dijitalleşme; öğretim, araştırma ve yönetim alanlarındaki süreçlerin dijital ortamda gerçekleştirilmesini içeren bir süreç olmakla beraber önemli bir rol oynamaktadır. Bu süreçte üniversiteler, dijital araçlar kullanarak öğrencilerin ve personelin deneyimlerini daha iyi hale getirmeyi amaçlamaktadır.

Bu makalede, Dokuz Eylül Üniversitesi İktisadi ve İdari Bilimler Fakültesi öğrenci işleri biriminin öğrencilerle iletişim kurması sürecinde herhangi bir yazılım bulunmaması çalışmanın problemi olarak tanımlanmıştır. Çalışmanın problemine yönelik bir destek yönetim sistemi geliştirmek amacıyla MEAN yığını kullanımı ele alınmıştır. Bu sistem, öğrenci işleri biriminin öğrenci sorunlarını sıkça sorulan sorular ve talep oluşturma yöntemleri ile çözmelerine olanak sağlayacak bir dizi araç ve özellikler sunmaktadır. Tam yığın geliştirme kavramının bir örneği olan MEAN; MongoDB, Express.js, Angular ve Node.js'yi içermektedir. Bu teknolojilerin birleşimi, bir web uygulaması geliştirmek için kullanılırken hepsinin JavaScript tabanlı olması daha hızlı ve daha verimli bir geliştirme süreci

sunmaktadır. Sistemin özellikleri arasında öğrenci taleplerinin yönetimi, sorun takibi ve yönetimi, öğrenci bilgilerinin ve verilerinin saklanması gibi temel özellikler yer almaktadır. Bu özellikler ile öğrenci işleri biriminin iş akışlarını yönetmelerine ve öğrencilerin sorunlarını daha hızlı ve etkili bir şekilde çözmelerine yardımcı olmak amaçlanmaktadır. Bu makalede bahsedilen teknolojilerin, web uygulaması geliştiricileri için faydalı olması ve bu alanda daha fazla çalışmaya yol açması beklenmektedir.

Gelecek çalışmalarda; makale kapsamına geliştirilen destek yönetim sistemi içerisinde işlem gören taleplerden elde edilen (oluşturulma tarih ve saat bilgisi, cevaplanma tarih ve saat bilgisi, öğrencinin bölüm bilgisi vb.) veriler kullanılarak birim personelinin kullanımına yönelik iş analitiği ekranı oluşturulması planlanmaktadır. Sistem içerisinde oluşturulan taleplerin durumunu sorgulamak amacıyla takip numarası atanması ve kullanıcıya bildirilmesi gelecek çalışmalarda ele alınacak diğer bir uygulamadır. Aynı zamanda yine gelecek çalışmaların konusu olarak; geliştirilen destek yönetim sisteminin DEÜ-İİBF öğrenci işleri biriminde görevli olan personel tarafından test edilmesi ve değerlendirilmesi planlanmaktadır.

KAYNAKLAR *(REFERENCES)*

[1] Y. Taşcı ve E. Taşlıbeyaz, "Yükseköğretim Kurumlarında Dijital Dönüşüm Çalışmalarının İncelenmesi," *Yükseköğretim ve Bilim Dergisi*, cilt 11, no. 1, ss. 172-183, Nisan 2021. Doi: 10.5961/jhes.2021.439

[2] A. Taşkıran, "Dijital Çağda Yükseköğretim, " *Açıköğretim Uygulamaları ve Araştırmaları Dergisi*, cilt 3, no. 1, ss. 96-109, Ocak 2017.

[3] E. K. Gümüşoğlu, "Yükseköğretimde Dijital Dönüşüm," *Açıköğretim Uygulamaları ve Araştırmaları Dergisi*, cilt 3, no. 4, ss. 30-42, Ekim 2017.

[4] Resmî Gazete, "Y. Kurumu, Yükseköğretim Üst Kuruluşları ile Yükseköğretim Kurumlarının İdari Teşkilatı Hakkında Kanun Hükmünde Kararname," *https://www.mevzuat.gov.tr*, 21 Kasım 1983. [Çevrimiçi]. Available: https://www.mevzuat.gov.tr/mevzuatmetin/4.5.124.pdf. [Erişildi: 20 Nisan 2023].

[5] N. Ayaz ve A. Arakaya, "Yükseköğretimde Hizmet Kalitesi Ölçümü: Öğrenci İşleri Daire Başkanlığı Örneği, " *Yükseköğretim ve Bilim Dergisi*, cilt 9, no. 1, ss. 123-133, Nisan 2019. Doi: 10.5961/jhes.2019.315

[6] E. Genç Kumtepe, E. Toprak, A. Öztürk, G. Tuna Büyükköse, H. Kılınç ve İ. Aydın Menderis, "Açık ve uzaktan öğrenmede destek hizmetleri: Yerelden küresele bir model," *Açıköğretim Uygulamaları ve Araştırmaları Dergisi*, cilt 5, no. 3, ss. 41-80, Temmuz 2019.

[7] A. Thoring, D. Rudolph, ve R. Vogl. "Digitalization of Higher Education from a Student's Point of View," *https://www.eunis.org*, 2017. [Çevrimiçi]. Available: https://www.eunis.org/download/2017/EUNIS_2017_paper_47.pdf. [Erişildi: 08 Nisan 2023].

[8] S. Aggarwal ve J. Verma, "Comparative Analysis of MEAN Stack and MERN Stack," *International Journal of Recent Research Aspects*, cilt 5, no. 1, ss. 127-132, Mart 2018.

[9] Y. Daşdemir ve B. C. Kara, "Farklı İş Yükleri Altında NoSQL Sistemlerinin Performans Analizi," *BEÜ Fen Bilimleri Dergisi*, cilt 8, no. 4, ss. 1466-1477, Aralık 2019. Doi: https://doi.org/10.17798/bitlisfen.547532

[10] V. Tecim. "Sistem Geliştirme Yaşam Döngüsü," *vahaptecim.com*, [Çevrimiçi]. Available: https://vahaptecim.com.tr/sistem-gelistirme-yasam-dongusu. [Erişildi: 12 Mart 2023].

[11] B. D. Dunka, E. A. Emmanuel ve D. O. Oyerinde, "Simplifying Web Asslication Development Using - Mean Stack," *International Journal of Latest Research in Engineering and Technology (IJLRET)*, cilt 04, no. 01, ss. 62-76, Ocak 2018.

[12] A. Taivalsaari, T. Mikkonen, C. Pautasso, ve K. Systä, "Full Stack Is Not What It Used to Be," *Lecture Notes in Computer Science*, ss. 363–371, Mayıs 2021. Doi: https://doi.org/10.1007/978-3-030-74296-6_28.

[13] N. Nirgudkar ve P. Singh, "The MEAN Stack," *International Research Journal of Engineering and Technology (IRJET)*, cilt 04, no. 05, ss. 3237-3239, Mayıs 2017.

[14] Angular Community, "What is Angular?," *https://angular.io/*, 2010. [Çevrimiçi]. Available: https://angular.io/guide/what-is-angular. [Erişildi: 12 Nisan 2023].

[15] M. A. Jadhav, B. R. Sawant ve A. Deshmukh, "Single Page Asslication using AngularJS," *International Journal of Computer Science and Information Technologies (IJCSIT)*, cilt 6, no. 3, ss. 2876-2879, 2015.

[16] P. Klauzinski ve J. Moore. *Mastering JavaScript Single Page Asslication Development*. Birmingham, UK: Packt Publishing Ltd., 2016.

[17] M. Karakoç, C. Ardıç ve M. A. E. Şen, "RESTful Web Servisleri ve Node.js Kullanılarak Genel Bir Kullanıcı Doğrulama Sisteminin Raspberry Pi ve RFID Teknolojisi ile Tasarımı ve Gerçekleştirimi," *İleri Mühendislik Çalışmaları ve Teknolojileri Dergisi*, cilt 2, no. 1, ss. 10-20, Haziran 2021.

[18] A. Mardan, *Practical Node.js: Using Express.js to Create Node.js Web Asss*. San Francisco, California, Apress, Berkeley, CA, 2018, ss. 51-87.

[19] B. De. *API Management: API Management*, Bangalore, Karnataka, India: Apress Berkeley, CA, 2017.

[20] E. Ayçin, "Veri Tabanı Yönetim Sistemi Seçiminde Swara ve Copras Yöntemlerinin Bütünleşik Olarak Kullanılması", *Journal of Business in The Digital Age*, cilt 1, no. 2, ss. 51-58, Aralık 2018.

[21] C. Huang, M. Cahill, A. Fekete ve U. Röhm, "Data Consistency Properties of Document Store as a Service (DSaaS): Using MongoDB Atlas as an Example," *Performance Evaluation and Benchmarking for the Era of Artificial Intelligence*, ss. 126-139, Ocak 2019. Doi: 10.1007/978-3-030-11404-6_10

[22] MongoDB Inc, "What is MongoDB Atlas?," *www.mongodb.com*, [Çevrimiçi]. Available: https://www.mongodb.com/docs/atlas/. [Erişildi: 23 Nisan 2023].

[23] S. Kurnaz, Ö. Çetin ve F. İnce, "Yazılım Mühendisliğinde Kalite ve UML," *Havacılık ve Uzay Teknolojileri Dergisi*, cilt 1, no. 2, ss. 1-12, Temmuz 2003.

[24] Microsoft Visual Studio Code, "Visual Studio" *www.microsoft.com*, [Çevrimiçi]. Available: https://visualstudio.microsoft.com/tr/#vscode-section. [Erişildi: 16 Nisan 2023].