

بسمه تعالی

دوره کارآموزی پروژه محور

پروژه : اتوماسیون اداری

تحت نظارت علیرضا بهمنش

نسخه 1.0.0

تابستان 1404

اکسل ارسالی از واحد بیژنس :

جدول کاربران / Users

Id	Username	Role	PasswordHash
u001	admin.reza	admin	\$2a\$12\$Xyz...adminHash
u002	fatemeh.staff	employee	\$2a\$12\$Abc...staffHash
u003	ali.approver	approver	\$2a\$12\$Pqr...approveHash
u004	mina.hr	hr	\$2a\$12\$Lmn...hrHash

نقش‌ها برای تعیین سطح دسترسی در گردش کار استفاده می شوند. رمزها هش شده هستند.

جدول فرآیند / WorkflowDefinitions

Id	Name	Description	CreatedAt	CreatedBy
wf001	LeaveApproval	فرآیند تأیید مرخصی ۲ مرحله‌ای	2024-06-01	u001
wf002	PurchaseRequest	فرآیند تأیید خرید تجهیزات	2024-06-12	u001

جدول ساختار مرحله به مرحله / WorkflowSteps

StepId	WorkflowId	Order	StepName	Role	Editable
ws01	wf001	1	بررسی توسط مدیر مستقیم	approver	true
ws02	wf001	2	تأیید منابع انسانی	hr	true
ws03	wf002	1	بررسی امور مالی	finance	true
ws04	wf002	2	تأیید مدیر کل	admin	false

📋 جدول درخواست ها / Requests

Id	Title	Description	CreatedByUserId	CreatedAt	CurrentStatus	CurrentStep	WorkflowId
r001	نیاز به مرخصی ۳ روزه جهت سفر خانوادگی	درخواست مرخصی تابستان	u002	2024-06-14	Level2Approved	2	wf001
r002	خرید مانیتور برای توسعه دهنده بک اند	درخواست خرید مانیتور	u002	2024-06-19	Pending	1	wf002

📋 جدول وضعیت هر مرحله تایید درخواست / ApprovalSteps

StepOrder	RequestId	ApproverUserId	Status	ApprovedAt
1	r001	u003	Approved	2024-06-15 09:20
2	r001	u004	Approved	2024-06-15 11:45
1	r002	u005 (finance)	Pending	null

📋 جدول لاگ های ورک فلو / WorkflowLogs

Id	RequestId	Step	ActionBy	ActionType	Timestamp
log001	r001	1	u003	Approve	2024-06-15T09:20:00
log002	r001	2	u004	Approve	2024-06-15T11:45:00
log003	r002	1	u005	Pending	2024-06-19T14:00:00

✿ جلسه اول: طراحی معماری پروژه، شناخت نیازها و پایه‌گذاری فنی

🎯 اهداف جلسه:

- درک کامل نیازهای اتوماسیون اداری در سازمان‌های واقعی
- تصمیم‌گیری درباره معماری Microservice و سرویس‌های مستقل
- تعریف نقش‌ها و مسیرهای دسترسی (Role Based Access)
- آماده‌سازی پروژه بک‌اند و فرانت‌اند برای توسعه در جلسات آینده

■ وظایف تیم بک‌اند

شرح کامل

تسک

؟
Re

طراحی دامنه‌ها

طراحی موجودیت‌های اصلی سیستم : User, Request, Workflow, ApprovalStep

ساخت پروژه بک‌اند

ایجاد پروژه ASP.NET Core Web API با EnterpriseAutomation.Api

ساخت لایه‌های
معماری

ایجاد لایه‌های Clean Domain, Application, Infrastructure برای Architecture

ایجاد کلاس‌های
ابتدایی

تعریف اولیه کلاس‌های User, Role, Request در لایه Domain

راه‌اندازی دیتابیس

نصب SQL Server یا Docker image + ساخت دیتابیس EnterpriseAutomation

■ وظایف تیم فرانت‌اند

شرح کامل

تسک

طراحی رابط‌های اولیه ساخت وایرفریم صفحات زیر در Figma یا کاغذ Dashboard: فرم درخواست، صفحه تأیید

ایجاد پروژه Next.js

ایجاد پروژه enterprise-automation-ui با TypeScript و ESLint

تنظیم ساختار پروژه

تعریف پوشه‌های pages, components, services, types, styles

تعریف مسیرهای
ابتدایی

تعریف صفحات /dashboard, /request/new, /request/[id] برای جلسات آینده

نوشتن صفحه اولیه

ساخت صفحه اصلی سیستم و درج متن خوش‌آمدگویی

جلسه دوم: آماده‌سازی محیط توسعه و اتصال سرویس‌ها

اهداف آموزشی:

- ساخت زیرساخت اولیه برای پروژه‌های بک‌اند و فرانت‌اند
- راه‌اندازی Docker Compose برای پایگاه‌های داده و سرویس‌های Keycloak
- ایجاد اتصال بین کانتینرها و تست صحت اجرای سرویس‌ها
- ساخت دیتابیس اصلی و تعریف اولیه Realm و Roles در Keycloak
- تنظیم پروژه فرانت‌اند برای اتصال به API ها و محیط توسعه محلی

وظایف تیم بک‌اند

تسک	شرح کامل
✓ 1. ساخت فایل docker-compose.yml	تعریف سرویس‌های sqlserver, keycloak, redis, backend-api و اتصال شبکه آنها
✓ 2. تعریف ایمیج SQL Server	ایجاد کانتینر با اعتبار اولیه، تنظیم volume برای داده‌ها
✓ 3. راه‌اندازی Keycloak	ساخت realm با نام enterprise-automation-client، ایجاد نقش‌های admin, employee, approver
✓ 4. ساخت دیتابیس و جدول اولیه	اتصال از API به SQL Server و اجرای migration اولیه با EF Core
✓ 5. تست اتصال سرویس‌ها	بررسی اینکه API به درستی به DB و Keycloak متصل است و توکن JWT تولید می‌شود

Post

Json

وظایف تیم فرانت‌اند

تسک	شرح کامل
1. اتصال به Keycloak	نصب و پیکربندی keycloak-js یا next-auth برای اتصال به Keycloak و دریافت توکن JWT
2. تست احراز هویت فرانت	اجرای ورود و دریافت اطلاعات کاربر از Realm تستی
3. تنظیم API Base URL	تعریف مسیر بک‌اند در فایل env.local برای ارتباط با سرور API داخل کانتینر
4. ساخت کامپوننت ورود اولیه	طراحی صفحه ورود ساده برای دریافت توکن از Keycloak و ذخیره در localStorage یا context در

جلسه سوم: مدل سازی کاربران و نقش ها + اتصال کامل به Keycloak

اهداف جلسه:

- ساخت جدول کاربران در دیتابیس و ارتباط با سیستم احراز هویت
- تعریف نقش ها به صورت Role-Based در Keycloak
- ساخت API ثبت و دریافت کاربران در بک اند
- دریافت اطلاعات کاربر وارد شده از Keycloak در فرانت اند
- نمایش صفحه پروفایل و تنظیمات پایه در UI

وظایف تیم بک اند

وظیفه	توضیح کامل
✓ طراحی مدل User	ساخت کلاس User در لایه Domain با فیلدهای ضروری
✓ ساخت جدول Users در دیتابیس	اجرای migration با EF Core برای ساخت جدول واقعی
ساخت سرویس ارتباط با Keycloak	ایجاد کلاس HttpClient برای خواندن اطلاعات کاربر و نقش از Keycloak
ساخت API ثبت یا خواندن کاربر	مسیر GET برای دریافت اطلاعات کاربر فعلی از JWT و مسیر POST برای ثبت اولیه

وظایف تیم فرانت اند

وظیفه	توضیح کامل
اتصال به Keycloak و استخراج اطلاعات	استفاده از توکن JWT برای دریافت نام، ایمیل و نقش کاربر از claims
ساخت صفحه پروفایل کاربر	نمایش نام، نقش، ایمیل، وضعیت فعالیت (Active/Inactive)
ساخت context یا hook برای مدیریت کاربر	مدیریت سراسری اطلاعات کاربر در کل پروژه و ارسال توکن ها در API ها
طراحی لینک های مناسب در navbar	مثل لینک "پروفایل من"، "خروج"، و نمایش نقش فعلی در header

جلسه چهارم: کنترل سطح دسترسی (Role-based Authorization)

اهداف این جلسه:

- محافظت مسیرهای API در بک‌اند بر اساس Role
- بررسی و تفسیر توکن JWT دریافتی از Keycloak
- محدودسازی کامپوننت‌ها و صفحات در فرانت بر اساس نقش کاربر
- پیاده‌سازی دسترسی دینامیک در UI برای کارمند، تأییدکننده و ادمین

وظایف تیم بک‌اند

شرح کامل	وظیفه
گرفتن role از Claim داخل توکن <u>JWT</u> صادرشده توسط Keycloak	1. تفسیر نقش از JWT
محدودسازی اکشن‌ها مثل تأیید، لیست، و حذف فقط برای نقش‌های مشخص	2. اعمال [Authorize] برای هر مسیر
بررسی اعتبارسنجی مسیرهای محدود با توکن‌های مختلف	3. تست توکن‌ها با Postman
پاسخ مناسب به کاربران بدون دسترسی مجاز	4. مدیریت خطاهای 403 و 401

وظایف تیم فرانت‌اند

شرح کامل	وظیفه
Parse کردن توکن و ذخیره role در context یا localStorage	1. استخراج نقش از JWT
فقط نمایش عناصر مربوط به نقش‌های مجاز	2. کنترل نمایش صفحات و دکمه‌ها
ساخت صفحه "403 دسترسی غیرمجاز"	3. هدایت کاربر بدون نقش به صفحه خطا
استفاده از شرط برای role در زمان render صفحات و اکشن‌ها	4. طراحی شرط‌های رندر کامپوننت

جلسه پنجم: طراحی و پیاده‌سازی بخش ثبت درخواست (Request)

اهداف آموزشی:

- مدل‌سازی موجودیت درخواست در پایگاه داده
- ساخت API ثبت درخواست جدید و بررسی اعتبار کاربر
- طراحی فرم ارسال درخواست در فرانت با اعتبارسنجی
- نمایش پیام‌های موفقیت یا خطا و ارتباط مؤثر با بک‌اند

وظایف تیم بک‌اند

شرح کامل	وظیفه
ساخت مدل Request در لایه Domain شامل عنوان، شرح، شناسه کاربر، زمان ایجاد، وضعیت	1. طراحی کلاس Request
اجرای Migration برای ذخیره جدول Requests	2. ساخت جدول در دیتابیس
دریافت داده از کاربر لاگین‌شده و ذخیره در دیتابیس	3. ساخت API POST برای ثبت درخواست
فقط کاربران با نقش employee مجاز به ثبت درخواست باشند	4. اعتبارسنجی توکن و نقش
در پاسخ به API ، ارسال RequestId یا پیام موفقیت	5. بازگردانی شناسه درخواست یا پیام موفق

وظایف تیم فرانت‌اند

شرح کامل	وظیفه
ورودی عنوان و شرح با اعتبارسنجی لازم	1. ساخت فرم ارسال درخواست
ارسال داده فرم با توکن کاربر واردشده	2. اتصال به API بک‌اند
استفاده از Toast یا Modal برای نمایش پیام‌ها	3. نمایش پیام موفقیت یا خطا
پاک‌سازی فیلدها و هدایت به صفحه لیست یا تأیید	4. ریست فرم پس از ارسال موفق
تست با داده‌های خالی، طول زیاد، یا ارسال دوباره	5. تست رفتار فرم

جلسه ششم: طراحی صفحه لیست درخواست‌ها + فیلتر دینامیک بر اساس نقش و وضعیت

اهداف آموزشی:

- طراحی API واکنشی درخواست‌ها در بک‌اند با فیلتر وضعیت و نقش
- پیاده‌سازی لیست درخواست‌ها در فرانت با نمایش خلاصه اطلاعات
- اعمال رنگ‌بندی، وضعیت مرحله فعلی، و دکمه‌های عملیات مرتبط
- پشتیبانی از فیلتر وضعیت (Pending, Approved...) و جستجوی متن

وظایف تیم بک‌اند

تسک	توضیح کامل
1. طراحی endpoint لیست درخواست‌ها	مسیر GET /api/request/list با فیلترهای optional: status, createdBy, role
2. خواندن role از JWT	بررسی نقش کاربر لاگین‌شده و ارسال داده مرتبط با سطح دسترسی
3. پشتیبانی از pagination و search	امکان ارسال query مثل <code>?status=Pending&page=2&search=</code>
4. بازگردانی داده مختصر	ارسال اطلاعات: Id, Title, CurrentStep, Status, CreatedAt

وظایف تیم فرانت‌اند

تسک	توضیح کامل
1. طراحی کامپوننت لیست درخواست‌ها	ساخت جدول با ستون‌های: عنوان، وضعیت، تاریخ، مرحله فعلی
2. نمایش رنگ وضعیت	رنگ‌آمیزی سلول یا Badge برای Pending, Approved, Rejected
3. افزودن فیلتر و جستجو	ساخت ورودی جستجو Dropdown + انتخاب وضعیت
4. رندر دکمه‌های عملیات	نمایش دکمه "مشاهده" برای همه و دکمه "تأیید" فقط برای approver در مرحله خود

جلسه هفتم: طراحی ساختار گردش کار (Workflow) و مراحل تأیید

اهداف آموزشی:

- تعریف مدل‌های WorkflowDefinition و WorkflowStep در دیتابیس
- طراحی ساختار مراحل تأیید با ترتیب و نقش مرتبط
- آماده‌سازی فرآیندها برای اتصال به درخواست‌ها
- نمایش بصری مراحل در فرانت با جزئیات مرحله به مرحله

وظایف تیم بک‌اند

توضیح کامل

مدل کلی یک فرآیند شامل نام، توضیح و ساختار مرحله‌ای JSON یا رابطه‌ای هر مرحله شامل ترتیب، نقش، نام مرحله و شرط (اختیاری) اجرای Migration برای ایجاد دو جدول اصلی مرتبط با گردش کار مسیر POST برای ثبت فرآیند جدید + مراحل مرتبط دریافت مراحل یک فرآیند بر اساس ID برای اتصال در جلسات آینده

تسک

1. طراحی مدل WorkflowDefinition
2. طراحی جدول WorkflowStep
3. پیاده‌سازی Migration پایگاه داده
4. ساخت API ایجاد گردش کار
5. طراحی Service واکنشی مراحل فرآیند

وظایف تیم فرانت‌اند

توضیح کامل

نمایش بصری مراحل تأیید مثل Stepper یا Timeline با ترتیب و نقش هر مرحله رنگ‌های خاص برای مراحل جاری، تأیید شده، و آینده فرم UI برای ثبت فرآیند جدید توسط مدیر شامل نام و مراحل با ترتیب و نقش ارسال داده مراحل به بک‌اند به صورت آرایه JSON یا

تسک

1. طراحی کامپوننت مراحل گردش کار
2. نمایش رنگ‌بندی مراحل
3. ساخت فرم تعریف گردش کار
4. اتصال به API ثبت گردش کار

✂ جلسه هشتم: اتصال درخواست‌ها به فرآیند گردش کار و تعیین مرحله فعلی

🎯 اهداف آموزشی:

- ثبت شناسه فرآیند (WorkflowId) در هر درخواست
- تعیین مرحله فعلی تأیید (CurrentStep) در مدل Request
- نگهداری وضعیت کلی درخواست (CurrentStatus)
- واکنشی مراحل مرتبط از گردش کار و ذخیره اولیه تأییدها
- نمایش مرحله در UI و آماده‌سازی برای عملیات تأیید در جلسات بعد

■ وظایف تیم بک‌اند

تسک	توضیح کامل
1. افزودن فیلد WorkflowId, CurrentStep, RequestStatus به مدل Request	در Migration جدید اضافه شود و فیلدهای پیش‌فرض تعیین شوند
2. واکنشی مراحل گردش کار هنگام ثبت Request	با استفاده از WorkflowId مراحل را واکنشی کرده و مرحله اول را تنظیم کنیم
3. ثبت مرحله اول به عنوان مرحله جاری	CurrentStep = 1 و CurrentStatus = "Pending" یا "Started" باشد
4. ذخیره تأییدات در جدول ApprovalSteps	مرحله اول را به صورت Pending برای تأییدکننده مرحله اول ثبت کنیم
5. بروزرسانی API ثبت درخواست	دریافت WorkflowId از ورودی فرم و انجام مراحل بالا به صورت کامل

■ وظایف تیم فرانت‌اند

تسک	توضیح کامل
1. بروزرسانی فرم ثبت درخواست	افزودن فیلد WorkflowId یا Dropdown انتخاب نوع فرآیند از API
2. واکنشی لیست فرآیندها از API	برای نمایش گزینه‌ها در فرم انتخاب
3. ارسال WorkflowId همراه با عنوان و شرح	ساخت body کامل شامل اطلاعات مورد نیاز بک‌اند
4. نمایش مرحله جاری در لیست درخواست‌ها	نمایش CurrentStep و نام مرحله در UI لیست درخواست‌ها

✂ جلسه نهم: ثبت لاگ تأیید درخواست‌ها و مانیتورینگ اقدامات کاربران

🎯 اهداف آموزشی:

- تعریف مدل لاگ عملیات (WorkflowLog) در بک‌اند
- ثبت همه اقدامات تأیید، رد، یا تغییر در مرحله درخواست
- ذخیره لاگ‌ها در دیتابیس یا سیستم‌هایی مانند MongoDB / Elasticsearch
- طراحی صفحه نمایش تاریخچه اقدامات در رابط کاربری
- افزایش شفافیت برای کاربران و مدیران در بررسی فرآیندها

■ وظایف تیم بک‌اند

شرح کامل	تسک
تعریف مدل عملیات شامل: نوع اقدام، نام کاربر، مرحله، زمان، شناسه درخواست	1. ساخت جدول WorkflowLog
در هر اکشن تأیید یا رد، یک رکورد جدید در جدول لاگ ثبت شود	2. ذخیره لاگ در عملیات تأیید
ذخیره نوع اقدام... Approve, Reject, Skip, Assign :	3. اضافه کردن ActionType
ذخیره لاگ‌ها در NoSQL برای گزارش‌گیری حجیم‌تر (/اختیاری)	4. پشتیبانی از MongoDB یا Elasticsearch
مسیر GET برای دریافت تاریخچه یک درخواست خاص با ترتیب زمانی	5. ساخت API دریافت لاگ‌ها

■ وظایف تیم فرانت‌اند

شرح کامل	تسک
نمایش مرحله، اقدام‌کننده، نوع اقدام و زمان انجام	1. طراحی کامپوننت تایم‌لاین اقدامات
فراخوانی request/{id}/logs و دریافت تاریخچه	2. اتصال به API لاگ
تبدیل UTC به زمان قابل خواندن (locale friendly)	3. رندر زمان‌های خوانا
هر نوع اقدام با رنگ یا آیکون متفاوت نمایش داده شود	4. رنگ‌بندی اقدامات
مثلاً  برای Approve ،  برای Reject ،  برای Pending	5. استفاده از آیکون‌ها یا نشان‌ها

🌿 جلسه دهم: کنترل دسترسی با نقش‌ها (Role-Based Authorization)

🎯 اهداف آموزشی این جلسه:

- اتصال بک‌اند به Keycloak برای بررسی نقش از توکن JWT
- محدودسازی دقیق API ها با استفاده از [Authorize(Roles = "...")]
- پیاده‌سازی حفاظت مسیرها در فرانت براساس نقش کاربر
- جلوگیری از نمایش دکمه‌ها و صفحات غیرمجاز در رابط کاربری
- آماده‌سازی ساختار برای تخصیص قابلیت‌ها به نقش‌های مختلف

■ وظایف تیم بک‌اند

تسک

توضیح کامل

1. راه‌اندازی JwtBearerAuthentication به Keycloak برای تأیید توکن JWT و استخراج Role
2. اعمال محدودیت در API ها استفاده از [Authorize(Roles = "roleName")] در کنترلرها برای محدود کردن دسترسی
3. خواندن اطلاعات از توکن استخراج شناسه کاربر و نقش از claims داخل توکن برای استفاده در سرویس‌ها
4. ساخت تست با Postman با توکن ارسال درخواست با نقش employee ، approver ، admin و تست موفقیت یا خطا 403 نقش‌های مختلف
5. ایجاد endpoint آزمایشی Role Protected ساخت یک مسیر ساده برای تست نقش مانند /api/test/admin-only

■ وظایف تیم فرانت‌اند

تسک

توضیح کامل

1. استخراج Role از توکن JWT تحلیل قسمت Payload از توکن JWT و ذخیره Role در localStorage یا context
2. نمایش یا پنهان‌سازی کامپوننت‌ها بر اساس Role فقط نمایش دکمه‌ها، منوها یا مسیرها برای نقش‌های مجاز (مثلاً دکمه تأیید برای approver)
3. ساخت صفحه خطا برای دسترسی غیرمجاز نمایش صفحه 403 با پیام "شما مجاز به دسترسی به این بخش نیستید"
4. ساخت کامپوننت کنترل شرطی برای نمایش فرزندان اگر نقش مجاز باشد ساخت کامپوننت RoleGuard
5. تست مسیرهای حساس با نقش‌های مختلف با استفاده از توکن‌های مختلف تست کنید که آیا رابط‌ها به درستی محدود شده‌اند یا نه

✂ جلسه یازدهم: نمایش رابط و مسیرهای مجاز بر اساس نقش کاربر

🎯 اهداف آموزشی:

- نمایش اطلاعات کامل کاربر لاگین شده در UI
- رندر صفحات، منوها و دکمه‌ها بر اساس نقش (employee, approver, admin)
- ارسال نقش و اطلاعات به فرانت از طریق API
- استفاده از Context و RoleGuard برای کنترل سطح دسترسی
- طراحی داشبورد مخصوص هر نقش با قابلیت‌های مجزا

■ وظایف تیم بک‌اند

شرح کامل	تسک
بازگرداندن نام کاربری، شناسه، نقش، ایمیل از توکن JWT یا Keycloak	1. ساخت API اطلاعات کاربر (/api/user/me)
خواندن claim های preferred_username, email, realm_access.roles	2. استخراج Role دقیق از JWT
تعریف UserDto با داده‌های موردنیاز در UI	3. ارسال مدل User به فرانت
بررسی نقش در اکشن‌هایی که فقط admin یا approver می‌تونن ببینن (مانند تعریف فرآیند)	4. محدودسازی مسیرهای خاص

■ وظایف تیم فرانت‌اند

شرح کامل	تسک
دریافت اطلاعات و ذخیره در Context یا useUser Hook	1. فراخوانی API /api/user/me پس از ورود
نمایش لینک‌ها و خلاصه‌ها بر اساس نوع نقش—برای مثال تأییدیه‌ها برای approver یا لیست فرآیندها برای admin	2. ساخت داشبورد مخصوص نقش
پنهان‌سازی یا فعال‌سازی دکمه‌ها، لینک‌ها و صفحات خاص	3. استفاده از RoleGuard برای کنترل رندر
نمایش نام خوش‌آمدگویی و نقش کاربر در navbar یا بالای صفحه	4. نمایش نام و نقش در Header
ایجاد نگرهان مسیرها برای جلوگیری از دسترسی مستقیم به صفحات غیرمجاز	5. ساخت Route Protection

✳ جلسه دوازدهم: پیاده‌سازی تأیید مرحله‌ای درخواست‌ها بر اساس نقش و ترتیب فرآیند

🎯 اهداف آموزشی:

- ایجاد API تأیید مرحله برای تأییدکننده مرحله جاری
- بررسی صحت نوبت مرحله و نقش کاربر هنگام تأیید
- ثبت تأیید در جدول ApprovalSteps و به‌روزرسانی CurrentStep در Request
- نمایش دکمه‌های تأیید یا رد در رابط کاربری بر اساس نقش و مرحله فعال
- ارسال وضعیت جدید به API و دریافت پاسخ به‌روزرسانی‌شده

■ وظایف تیم بک‌اند

توضیح	تسک
مسیر POST /api/request/{id}/approve با بررسی نقش، نوبت، و ثبت عملیات فقط اگر مرحله فعلی با CurrentStep برابر باشد، مجاز به اقدام است	1. ساخت اکشن تأیید مرحله در ApprovalController
تغییر Status از Pending به Approved برای مرحله فعلی در صورت وجود مرحله بعدی، انتقال به آن؛ در غیراین‌صورت وضعیت کلی را Approved کن	2. بررسی نوبت مرحله
با استفاده از WorkflowLog، ثبت تأیید در تاریخ، مرحله، کاربر تأییدکننده	3. ثبت عملیات در ApprovalSteps
	4. افزایش CurrentStep در Request
	5. ثبت لاگ اقدام

■ وظایف تیم فرانت‌اند

توضیح	تسک
بررسی request.currentStep و مقایسه با نقش کاربر	1. نمایش دکمه "تأیید" فقط در مرحله فعال و نقش مجاز
ارسال شناسه درخواست در مسیر، و نمایش پیام موفقیت یا خطا	2. اتصال دکمه به API POST /approve
به‌روزرسانی مرحله جاری، وضعیت کلی، و رنگ‌بندی مرحله‌ها	3. رفرش لیست یا جزئیات پس از تأیید
غیرفعال‌سازی دکمه پس از اقدام موفق برای جلوگیری از دوباره‌کاری	4. جلوگیری از تکرار تأیید
اعلان موفقیت، خطا یا عدم مجوز برای نقش‌های نامرتب	5. نمایش Toast یا Modal پاسخ

جلسه سیزدهم: پیاده سازی گردش کار چند مرحله ای داینامیک

اهداف جلسه:

- مدیریت تأیید مرحله ای از چند مرحله تعریف شده
- بررسی وجود مرحله بعدی و تعیین فعال شدن آن
- به روز رسانی وضعیت تأیید مراحل در جدول ApprovalSteps
- به روز رسانی CurrentStep و CurrentStatus در Request
- نمایش مراحل تأیید شده و فعال در رابط کاربری با رنگ بندی و نقش مرتبط

وظایف تیم بک اند

توضیحات

تسک

1. بررسی وجود مرحله بعدی
پس از تأیید مرحله فعلی، بررسی شود آیا مرحله بعدی در جدول WorkflowSteps وجود دارد
2. انتقال مرحله
اگر مرحله بعدی وجود داشت، به روز رسانی CurrentStep و ثبت رکورد جدید در ApprovalSteps با نقش مرحله بعد
3. تغییر وضعیت نهایی
اگر مرحله فعلی آخرین مرحله بود، تغییر CurrentStatus به Approved و توقف گردش کار
4. ثبت لاگ حرکت بین مراحل
ثبت عملیات Step Transition یا Completed در WorkflowLogs
5. ارسال اطلاعات مرحله فعال
در پاسخ API، ارسال مرحله فعال بعدی و نقش مربوطه برای نمایش در فرانت

وظایف تیم فرانت اند

توضیحات

تسک

1. نمایش رنگ مراحل
مرحله تأیید شده = سبز، مرحله فعال = آبی، مراحل آینده = 1. نمایش رنگ مراحل خاکستری
2. نمایش نقش تأیید کننده مرحله جاری
نمایش نقش مورد نیاز برای مرحله جاری برای کاربر فعلی
3. مخفی سازی دکمه تأیید برای مراحل غیر مجاز
اگر نقش کاربر با نقش مرحله جاری همخوانی نداشت، دکمه ها غیر فعال باشند
4. نمایش وضعیت نهایی
اگر CurrentStatus == Approved بود، نمایش پیام "پایان گردش تأیید"
5. به روز رسانی رابط پس از تأیید
واکشی مجدد اطلاعات درخواست و نمایش مرحله جدید فعال شده

🌿 جلسه چهاردهم: ارتباط غیرهمزمان بین سرویس‌ها (RabbitMQ/gRPC) + نوتیفیکیشن تأیید

🎯 اهداف آموزشی:

- راه‌اندازی و اتصال RabbitMQ برای ارسال و دریافت پیام بین سرویس‌ها
- ارسال رویداد تأیید مرحله از سرویس Approval به سرویس Notification
- پیاده‌سازی Consumer برای دریافت پیام و ارسال نوتیفیکیشن یا ایمیل
- طراحی API برای وضعیت لحظه‌ای درخواست‌ها
- نمایش اعلان تأیید در فرانت پس از انجام عملیات بدون بارگذاری صفحه

■ وظایف تیم بک‌اند

تسک

توضیح کامل

- | | |
|---|--|
| 1. نصب و پیکربندی RabbitMQ | افزودن سرویس RabbitMQ به docker-compose.yml و نصب کتابخانه‌های MassTransit یا Raw RabbitMQ Client در پروژه |
| 2. ساخت Producer | در لحظه تأیید مرحله، ارسال پیام شامل RequestId, Step, ApproverId, Timestamp به صف نوتیفیکیشن |
| 3. ساخت Consumer در سرویس NotificationService | دریافت پیام، ذخیره نوتیفیکیشن یا ارسال ایمیل تستی |
| 4. ساخت کلاس‌های MessageContract | تعریف مدل پیام بین سرویس‌ها مثل StepApprovedMessage و استفاده در ارسال و دریافت |
| 5. تست با پیام آزمایشی | ارسال دستی پیام تأیید مرحله و بررسی دریافت توسط Consumer |

■ وظایف تیم فرانت‌اند

تسک

توضیح کامل

- | | |
|--------------------------------|--|
| 1. ساخت سیستم polling ساده | به صورت دوره‌ای (مثلاً هر 30 ثانیه) وضعیت مرحله جاری درخواست را بررسی کند |
| 2. طراحی نوتیفیکیشن UI | نمایش پیام "مرحله تأیید شد" پس از دریافت وضعیت جدید، با استفاده از Modall یا Toast |
| 3. ساخت کامپوننت وضعیت لحظه‌ای | نشان دادن مرحله فعلی بدون نیاز به refresh صفحه |
| 4. مدیریت حالت loading / جدید | نمایش spinner هنگام انتظار و اعلان در صورت تغییر وضعیت |
| 5. طراحی fallback در صورت خطا | نمایش پیام خطا یا عدم دسترسی در صورت عدم پاسخ API یا دریافت پیام نوتیفیکیشن |

✪ جلسه پانزدهم: تست واحد (Unit Test) و تست رابط کاربری (UI Test)

🎯 اهداف آموزشی:

- پوشش تست برای سرویس‌ها، کنترلرها و مدل‌ها در بک‌اند
- تست فرم‌ها، کامپوننت‌ها و رفتار کاربر در فرانت‌اند
- تضمین صحت عملکرد مسیرهای حساس (مثل تأیید مرحله)
- کاهش ریسک تغییرات آینده با تست‌های خودکار
- آماده‌سازی برای CI/CD و بررسی اتوماتیک Pull Request ها

■ وظایف تیم بک‌اند

شرح کامل	تسک
ساخت پروژه EnterpriseAutomation.Tests برای بک‌اند	1. راه‌اندازی پروژه تست با xUnit
تست تابع‌هایی مثل MarkApproved() و MockGetWorkflowSteps()	2. نوشتن Unit Test برای سرویس‌ها
ارسال درخواست واقعی به API و بررسی پاسخ و تغییرات دیتابیس	3. نوشتن Integration Test برای API
اجرای تست‌های شبیه‌سازی‌شده با راه‌اندازی اپلیکیشن واقعی در حافظه	4. استفاده از WebApplicationFactory
سنجش درصد تست‌شدگی کلاس‌ها، سرویس‌ها و مسیرهای مهم	5. بررسی پوشش تست با ابزار Coverlet

■ وظایف تیم فرانت‌اند

شرح کامل	تسک
راه‌اندازی Jest ، React Testing Library و MSW برای Mock API	1. نصب کتابخانه‌های تست
تست Form ثبت درخواست: مقداردهی، تعامل، ارسال داده، و نمایش پیام موفقیت	2. نوشتن تست فرم‌ها
بررسی ظاهر دکمه‌ها، تغییر رنگ مراحل، و وضعیت شرطی در ApproveButton	3. نوشتن تست کامپوننت‌های حالت
بررسی تغییرات ساختار JSX در کامپوننت‌ها و هشدار در تغییرات ناخواسته	4. استفاده از Snapshot تست
تست تأخیر در دریافت داده، حالت Loading ، و نمایش نوتیفیکیشن بعد از تأیید	5. بررسی رفتار async

جلسه شانزدهم Docker: استقرار در + Azure پیاده سازی CI/CD

اهداف آموزشی:

- ساخت Dockerfile برای سرویس های بک اند و فرانت اند
- تنظیم docker-compose.yml برای اجرای کامل پروژه
- اتصال پروژه به Azure Container Registry و Azure App Service
- راه اندازی Pipeline برای Build و Deploy خودکار
- تست نهایی رابط کاربری در محیط واقعی با API های مستقر

وظایف تیم بک اند

توضیح	تسک
مشخص کردن Base image ، مراحل Restore ، Build و Publish	1. ساخت Dockerfile برای پروژه API
اتصال سرویس ها (API) ، Redis ، SQL Server ، Keycloak ، RabbitMQ	2. تنظیم فایل docker-compose.yml کامل
ارسال تصویر ساخته شده به Azure با دستور docker push	3. اتصال به Azure Container Registry
اجرای خودکار Build و انتشار API در Azure App Service	4. راه اندازی Pipeline با Azure DevOps یا GitHub Actions
بررسی مسیرهای تولید مثل /api/request, curl یا Postman با /api/user/me	5. تست API مستقر

وظایف تیم فرانت اند

توضیح	تسک
استفاده از Base Image رسمی، نصب پکیج ها، اجرای Build	1. ساخت Dockerfile برای پروژه Next.js
اتصال رابط به مسیر واقعی API در Azure (https://api.enterprise.com)	2. تنظیم مسیرهای API در .env.production
ساخت تصویر فرانت اند و ارسال به Azure	3. Build پروژه با docker build و Push به ACR
فعال سازی تنظیمات real token ، مسیرهای دقیق، logging مناسب	4. پیاده سازی پشتیبانی از Production Mode
بررسی عملکرد فرم ها، لیست درخواست ها، نقش ها، و دکمه ها	5. تست رابط در محیط مستقر

جلسه هفدهم: کش کردن فرآیند گردش کار با Redis برای افزایش سرعت

اهداف آموزشی:

- ذخیره ساختار فرآیندهای گردش کار در Redis به جای خواندن مکرر از دیتابیس
- طراحی سرویس CacheService برای واکشی، ذخیره و اعتبارسنجی داده‌های کش‌شده
- به‌روزرسانی فرآیندها با قابلیت Refresh Cache
- اتصال سرویس گردش کار به کش هنگام فراخوانی مراحل
- طراحی رابط مدیریتی برای کنترل دستی کش از سمت مدیر

وظایف تیم بک‌اند

توضیحات	تسک
افزودن سرویس Redis در docker-compose.yml و تنظیمات اتصال در API	1. راه‌اندازی Redis در پروژه
نوشتن متدهایی برای GetWorkflowDefinition, SetWorkflowDefinition, RefreshWorkflow	2. طراحی کلاس CacheService
اگر فرآیند موجود بود → خواندن از Redis؛ اگر نبود → خواندن از DB و ذخیره در کش	3. بررسی موجود بودن فرآیند در کش
تعیین زمان اعتبار هر فرآیند ذخیره‌شده (مثلاً ۱۰ دقیقه یا ۲۴ ساعت)	4. تعریف TTL زمان انقضا
API /api/workflow/{id}/refresh توسط admin برای به‌روزرسانی کش فقط	5. طراحی مسیر Refresh دستی

وظایف تیم فرانت‌اند

توضیحات	تسک
هنگام مشاهده جزئیات درخواست، مراحل از API کش‌شده فراخوانی شود	1. نمایش سریع مراحل گردش کار
در پنل مدیر، امکان به‌روزرسانی فرآیند به صورت دستی ایجاد شود	2. طراحی دکمه "Refresh Cache"
نمایش LastRefreshed یا پیام "فرآیند به‌روزرسانی شد" پس از اقدام	3. نمایش زمان آخرین به‌روزرسانی
تست تغییر بصری پس از دریافت داده جدید از API	4. تست تغییر مراحل پس از Refresh

جلسه هجدهم: نمایش وضعیت دقیق یک درخواست خاص



اهداف آموزشی:

- ساخت API برای دریافت وضعیت کامل یک درخواست
- نمایش مرحله فعلی، مراحل تأیید شده، کاربران تأییدکننده و زمان انجام هر مرحله
- طراحی رابط کاربری برای نمایش اطلاعات به صورت تایملاین یا خلاصه وضعیت
- جلوگیری از فراخوانی چندگانه داده‌ها با ترکیب اطلاعات لاگ و مرحله فعلی
- فراهم‌سازی گزارش‌گیری تک‌درخواستی برای تیم منابع انسانی یا مدیران

وظایف تیم بک‌اند

توضیح	تسک
بازگرداندن اطلاعات دقیق مرحله جاری، کل فرآیند، و مرحله‌های تأیید شده	1. ساخت endpoint /request/{id}/status
دریافت نام مرحله فعلی، نقش موردنیاز، و تاریخچه اقدامات	2. ترکیب اطلاعات از Request, WorkflowStep, ApprovalSteps, WorkflowLogs
استفاده از DTO شامل stepNumber, stepName, role, status, approvedBy, approvedAt	3. تبدیل داده‌ها به مدل خوانا برای فرانت
اگر گردش کار به پایان رسیده، درج "Approved" یا "Rejected" به صورت واضح	4. ارسال وضعیت نهایی
اگر قبلاً واکنشی شده بود، اطلاعات از Redis کش دریافت شود برای افزایش عملکرد	5. کش‌سازی اطلاعات

وظایف تیم فرانت‌اند

توضیح	تسک
نمایش مراحل تأییدشده، مرحله فعال، و مراحل آینده به صورت تایملاین یا کارت	1. طراحی کامپوننت وضعیت درخواست
مثلاً <input checked="" type="checkbox"/> سبز برای "Approved"،  خاکستری برای "Pending"،  قرمز برای "Rejected"	2. استفاده از رنگ و آیکون برای وضعیت‌ها
نمایش جزئیات مربوط به هر مرحله در بخش اطلاعات	3. نمایش نام تأییدکننده و زمان اقدام
پس از تأیید کاربر، UI وضعیت به‌روزرسانی شود	4. بروزرسانی خودکار بعد از تأیید
ظاهر مناسب در دسکتاپ و موبایل برای نمایش مراحل و اطلاعات	5. طراحی Responsive

✂ جلسه نوزدهم: طراحی نمای تصویری گردش کار + تاریخچه تأییدات

🎯 اهداف این جلسه:

- نمایش ساختار کامل فرآیند به صورت گراف تصویری (Workflow Map)
- طراحی پنل دوبخشی: سمت چپ برای مرحله‌ها، سمت راست برای اقدامات انجام‌شده
- فراهم‌سازی قابلیت ویرایش مراحل برای مدیر سیستم
- نمایش تاریخچه تأییدها با جزئیات کاربر، مرحله و زمان
- افزایش دید مدیریتی برای تحلیل تأخیرها و اصلاح فرآیند

■ وظایف تیم بک‌اند

تسک	توضیح کامل
1. ساخت API GET /workflow/{id}/map	بازگرداندن ساختار مراحل فرآیند شامل نام مرحله، نقش، ترتیب، و اتصال بین مراحل
2. ساخت API POST /workflow/{id}/update	دریافت مرحله‌های جدید با ترتیب و نقش برای ویرایش فرآیند توسط admin
3. ساخت API دریافت تأییدهای درخواست	مسیر request/{id}/approvals شامل لیست مراحل، کاربر تأییدکننده، وضعیت و زمان
4. ثبت و بروزرسانی در DB + Refresh Cache	بعد از ویرایش، ذخیره مراحل جدید در DB و به‌روزرسانی Redis Cache
5. محدودسازی API ویرایش فقط برای admin	بررسی [Authorize(Roles = "admin")] در مسیرهای ویرایشی

■ وظایف تیم فرانت‌اند

تسک	توضیح کامل
1. طراحی بخش چپ رابط برای نمایش Map مراحل گردش کار	استفاده از نمودار Mermaid یا ساختار Box و Arrow برای نمایش بصری مراحل
2. نمایش پنل سمت راست برای تاریخچه تأییدات	لیست مراحل انجام‌شده، نام کاربر، وضعیت و زمان با رنگ‌بندی مناسب
3. ساخت فرم ویرایش مراحل (فقط مدیر)	قابلیت Drag & Drop، ویرایش نام مرحله و نقش هر مرحله
4. ارسال داده جدید به API بروزرسانی	در صورت تغییر، ارسال لیست مرحله‌ها به بک‌اند با تأیید نهایی
5. واکنشی مجدد داده‌ها پس از تغییر	نمایش مرحله‌های جدید + هشدار بروزرسانی موفق

**با تشکر از همراهی شما
پرتلاش و موفق باشید**