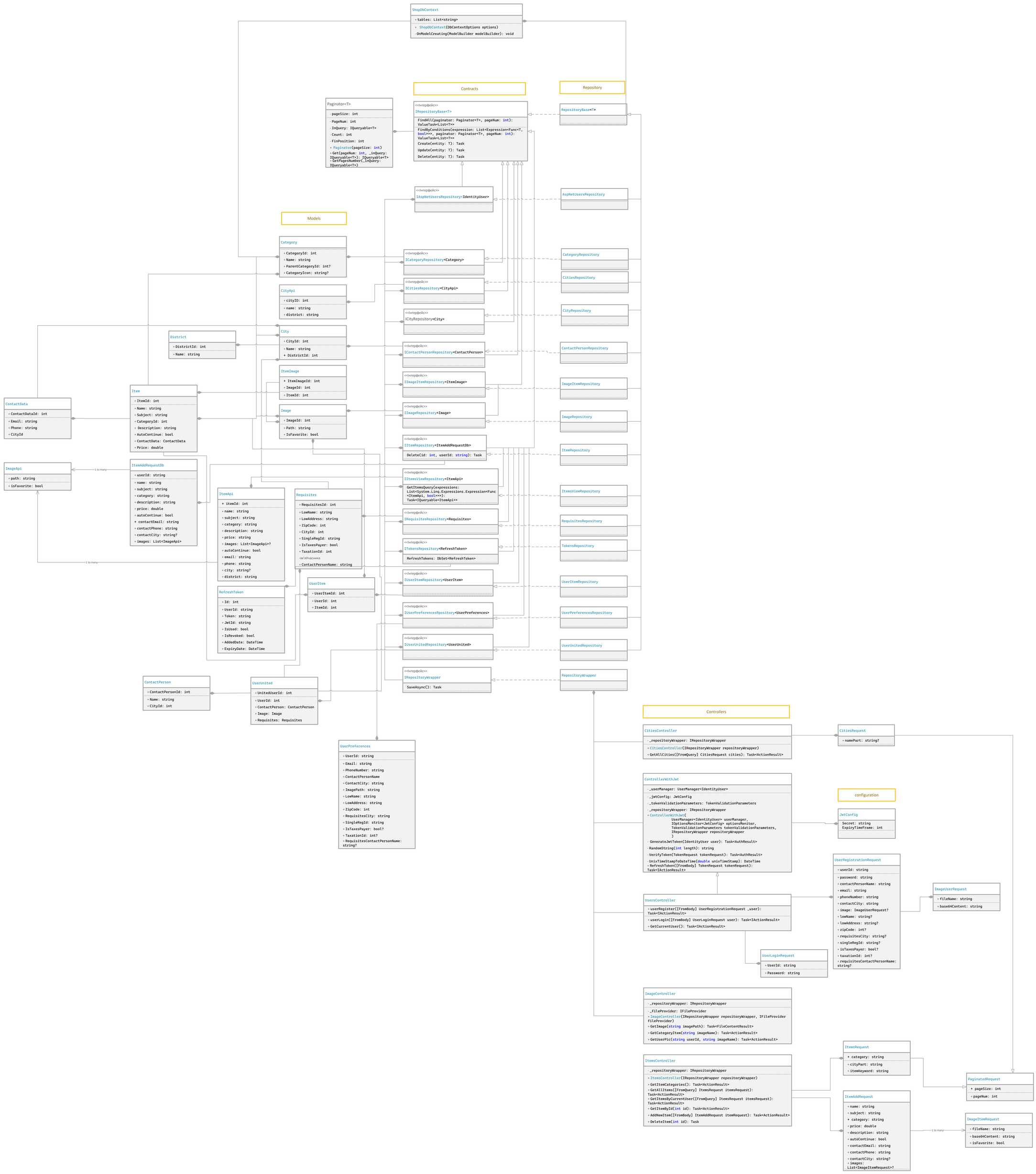
ShopDbContext
- tables: List<string>
+ ShopDbContext(DbContextOptions options)
- OnModelCreating(ModelBuilder modelBuilder): void
Contracts
Repository
Paginator<T>
- pageSize: int
- PageNum: int
- InQuery: IQueryable<T>
- Count: int
- FinPosition: int
+ Paginator(pageSize: int)
+ Get(pageNum: int, _inQuery: IQueryable<T>): IQueryable<T>
+ GetPagesNumber(_inQuery: IQueryable<T>)
<<интерфейс>>
IRepositoryBase<T>
FindAll(paginator: Paginator<T>, pageNum: int): ValueTask<List<T>>
FindByConditions(expression: List<Expression<Func<T, bool>>>, paginator: Paginator<T>, pageNum: int): ValueTask<List<T>>
Create(entity: T): Task
Update(entity: T): Task
Delete(entity: T): Task
RepositoryBase<T>
<<интерфейс>>
IAspNetUsersRepository<IdentityUser>
AspNetUsersRepository
Models
Category
+ CategoryId: int
+ Name: string
+ ParentCategoryId: int?
+ CategoryIcon: string?
<<интерфейс>>
ICategoryRepository<Category>
CategoryRepository
CityApi
+ cityID: int
+ name: string
+ district: string
<<интерфейс>>
ICitiesRepository<CityApi>
CitiesRepository
<<интерфейс>>
ICityRepository<City>
CityRepository
District
+ DistrictId: int
+ Name: string
City
+ CityId: int
+ Name: string
+ DistrictId: int
<<интерфейс>>
IContactPersonRepository<ContactPerson>
ContactPersonRepository
ItemImage
+ ItemImageId: int
+ ImageId: int
+ ItemId: int
<<интерфейс>>
IImageItemRepository<ItemImage>
ImageItemRepository
Item
+ ItemId: int
+ Name: string
+ Subject: string
+ CategoryId: int
+ Description: string
+ AutoContinue: bool
+ ContactData: ContactData
+ Price: double
Image
+ ImageId: int
+ Path: string
+ IsFavorite: bool
<<интерфейс>>
IImageRepository<Image>
ImageRepository
ContactData
+ ContactDataId: int
+ Email: string
+ Phone: string
+ CityId
<<интерфейс>>
IItemRepository<ItemAddRequestDb>
Delete(id: int, userId: string): Task
ItemRepository
ImageApi
+ path: string
+ isFavorite: bool
1 to many
ItemAddRequestDb
+ userId: string
+ name: string
+ subject: string
+ category: string
+ description: string
+ price: double
+ autoContinue: bool
+ contactEmail: string
+ contactPhone: string
+ contactCity: string?
+ images: List<ImageApi>
<<интерфейс>>
IItemsViewRepository<ItemApi>
GetItemsQuery(expressions: List<System.Linq.Expressions.Expression<Func<ItemApi, bool>>>): Task<IQueryable<ItemApi>>
ItemsViewRepository
ItemApi
+ itemId: int
+ name: string
+ subject: string
+ category: string
+ description: string
+ price: string
+ images: List<ImageApi>?
+ autoContinue: bool
+ email: string
+ phone: string
+ city: string?
+ district: string
Requisites
+ RequisitesId: int
+ LowName: string
+ LowAddress: string
+ ZipCode: int
+ CityId: int
+ SingleRegId: string
+ IsTaxesPayer: bool
+ TaxationId: int
-ім'ятчасника
+ ContactPersonName: string
<<интерфейс>>
IRequisitesRepository<Requisites>
RequisitesRepository
<<интерфейс>>
ITokensRepository<RefreshToken>
RefreshTokens: DbSet<RefreshToken>
TokensRepository
RefreshToken
+ Id: int
+ UserId: string
+ Token: string
+ JwtId: string
+ IsUsed: bool
+ IsRevoked: bool
+ AddedDate: DateTime
+ ExpiryDate: DateTime
UserItem
+ UserItemId: int
+ UserId: int
+ ItemId: int
<<интерфейс>>
IUserItemRepository<UserItem>
UserItemRepository
<<интерфейс>>
IUserPreferencesRpository<UserPreferences>
UserPreferencesRepository
<<интерфейс>>
IUserUnitedRepository<UserUnited>
UserUnitedRepository
<<интерфейс>>
IRepositoryWrapper
SaveAsync(): Task
RepositoryWrapper
ContactPerson
+ ContactPersonId: int
+ Name: string
+ CityId: int
UserUnited
+ UnitedUserId: int
+ UserId: int
+ ContactPerson: ContactPerson
+ Image: Image
+ Requisites: Requisites
UserPreferences
+ UserId: string
+ Email: string
+ PhoneNumber: string
+ ContactPersonName
+ ContactCity: string
+ ImagePath: string
+ LowName: string
+ LowAddress: string
+ ZipCode: int
+ RequisitesCity: string
+ SingleRegId: string
+ IsTaxesPayer: bool?
+ TaxationId: int?
+ RequisitesContactPersonName: string?
Controllers
CitiesController
- _repositoryWrapper: IRepositoryWrapper
+ CitiesController(IRepositoryWrapper repositoryWrapper)
+ GetAllCities([FromQuery] CitiesRequest cities): Task<ActionResult>
CitiesRequest
+ namePart: string?
ControllerWithJwt
- _userManager: UserManager<IdentityUser>
- _jwtConfig: JwtConfig
- _tokenValidationParameters: TokenValidationParameters
- _repositoryWrapper: IRepositoryWrapper
+ ControllerWithJwt(UserManager<IdentityUser> userManager, IOptionsMonitor<JwtConfig> optionsMonitor, TokenValidationParameters tokenValidationParameters, IRepositoryWrapper repositoryWrapper)
- GenerateJwtToken(IdentityUser user): Task<AuthResult>
- RandomString(int length): string
- VerifyToken(TokenRequest tokenRequest): Task<AuthResult>
- UnixTimeStampToDateTime(double unixTimeStamp): DateTime
+ RefreshToken([FromBody] TokenRequest tokenRequest): Task<IActionResult>
configuration
JwtConfig
Secret: string
ExpiryTimeFrame: int
UsersController
+ userRegister([FromBody] UserRegistrationRequest _user): Task<IActionResult>
+ userLogin([FromBody] UserLoginRequest user): Task<IActionResult>
+ GetCurrentUser(): Task<IActionResult>
UserRegistrationRequest
+ userId: string
+ password: string
+ contactPersonName: string
+ email: string
+ phoneNumber: string
+ contactCity: string
+ image: ImageUserRequest?
+ lowName: string?
+ lowAddress: string?
+ zipCode: int?
+ requisitesCity: string?
+ singleRegId: string?
+ isTaxesPayer: bool?
+ taxationId: int?
+ requisitesContactPersonName: string?
ImageUserRequest
+ fileName: string
+ base64Content: string
UserLoginRequest
+ UserId: string
+ Password: string
ImageController
- _repositoryWrapper: IRepositoryWrapper
- _fileProvider: IFileProvider
+ ImageController(IRepositoryWrapper repositoryWrapper, IFileProvider fileProvider)
+ GetImage(string imagePath): Task<FileContentResult>
+ GetCategoryItem(string imageName): Task<ActionResult>
+ GetUserPic(string userId, string imageName): Task<ActionResult>
ItemsController
- _repositoryWrapper: IRepositoryWrapper
+ ItemsController(IRepositoryWrapper repositoryWrapper)
+ GetItemCategories(): Task<ActionResult>
+ GetAllItems([FromQuery] ItemsRequest itemsRequest): Task<ActionResult>
+ GetItemsByCurrentUser([FromQuery] ItemsRequest itemsRequest): Task<ActionResult>
+ GetItemById(int id): Task<ActionResult>
+ AddNewItem([FromBody] ItemAddRequest itemRequest): Task<ActionResult>
+ DeleteItem(int id): Task
ItemsRequest
+ category: string
+ cityPart: string
+ itemKeyword: string
PaginatedRequest
+ pageSize: int
+ pageNum: int
ItemAddRequest
+ name: string
+ subject: string
+ category: string
+ price: double
+ description: string
+ autoContinue: bool
+ contactEmail: string
+ contactPhone: string
+ contactCity: string?
+ images: List<ImageItemRequest>?
1 to many
ImageItemRequest
+ fileName: string
+ base64Content: string
+ isFavorite: bool