**Razredni diagrami**

**Kontoleri**

Kontroleri služe za dohvat odgovarajućih podataka iz modela te prikazivanje pogleda

povezanih uz danu akciju, u projektu imamo slijedeće kontrolere:

* **Application controller**

Glavni kontroler cijele aplikacije koji prikazuje layout u kojem se prikazuju ostali kontroleri. Korisnici i posjetitielji ne vide niti mogu utjecati na proces izvođenja ovog kontrolera, on sadrži metode koje moraju biti globalno dostupne. Osnovne funkcionalnosti ovog modela su: da provjerava dali je korisnik prijavljen u sustav, vraća trenutnog korisnika, ispisuje HTML tagove za Faviconu i za ikonu za socijalne mreže, dohvaća URL gravatar slike trenutnog korisnika I vraća puni naslov stranice.

* **Users controller**

Ovaj kontroler sadrži metode za stvaranje, brisanje, izmjenu I pregled korisnika u sustavu. On omogućuje postojećem korisniku, ako ima privilegije za to, da stvori, izmijeni ili pregleda profil drugog korisnika.

* **Categories controller**

Ovaj kontroler sadrži metode za stvaranje, brisanje, izmjenu i pregled kategorija.

On kao ontroler kategorija ima prije svega kozmetičku svrhu, namjena mu je da

omogući grupiranje stranica u kategorije i registriranom korisniku nudi mogućnosti editiranja, brisanja i stavaranja novih kategorija, te prikaz stranica unutar neke kategorije. Pošto se kategorije obično organiziraju u strukture stabla model kategorija sadrži polje “ancestry” u koje ancestry gem upisuje organizaciju

stabla.

* **Networks controller**

Ovaj kontroler sadrži metode za stvaranje, brisanje, izmjenu socijalnih mreža, on nema funkcionalnu, već kozmetičku svrhu te omogućuje dodavanje poveznica na socijalne mreže smao ovlaštenim korisnicima.

* **Statements controller**

Ovaj kontroler sadrži metode za stvaranje, brisanje, izmjenu i pregled prometa.

On nudi opciju prikaza dnevnog, mjesečnog i godišnjeg prometa ovlaštenom

korisniku.

* **Parts controller**

Sadrži metode za stvaranje, brisanje, izmjenu i pregled pojedinih dijelova.

* **Working Hours controller**

Sadrži metode za stvaranje, brisanje, izmjenu radnog vremena zaposlenika.

* **Roles controller**

Sadrži metode za stvaranje, brisanje, izmjenu uloga pojedinih osoba koje rade za firmu.

* **Sessions controller**

Sadrži metode za stvaranje i brisanje session kolačića u koji je upisan

autentifikacijski ključ.

* **Clients controller**

Sadrži metode za stvaranje, brisanje, izmjenu i pregled klijenata.

* **Procedures controller**

Sadrži metode za stvaranje, brisanje, izmjenu i pregled postupaka.

* **Pages controller**

Sadrži metode za stvaranje, brisanje, izmjenu i pregled stranica te nudi opciju pretraživanja stranica po njihovim tagovima. Dodatno još nudi opciju prikaza sakrivenih stranica i neobjavljenih stranica.

* **Settings controller**

Sadrži metode za izmjenu postavki.

* **Orders controller**

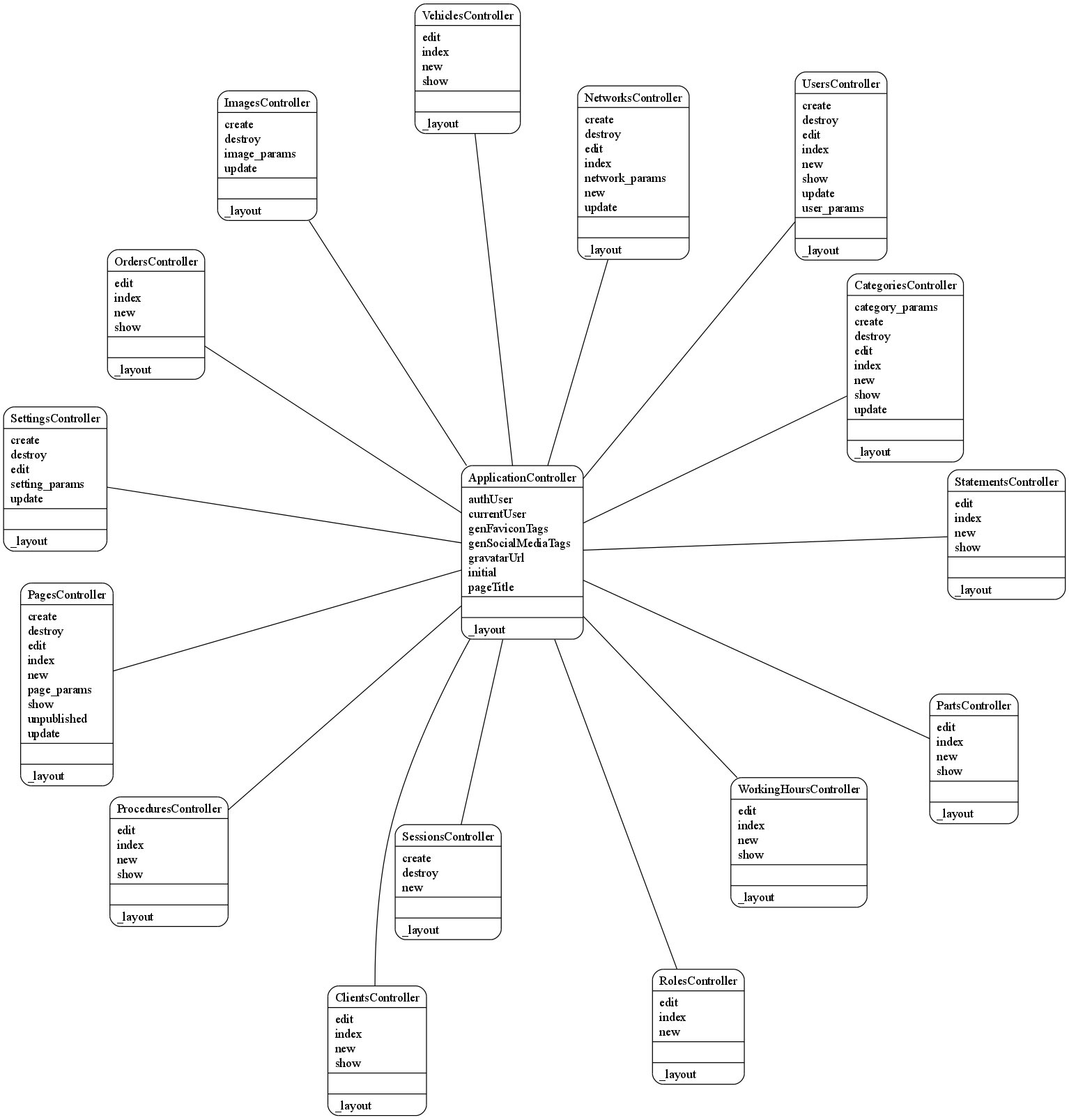
Sadrži metode za stvaranje, brisanje, izmjenu i pregled narudžbi.

* **Images controller**

Sadrži metode za stvaranje, brisanje, izmjenu slika.

* **Vehicles controller**

Sadrži metode za stvaranje, brisanje, izmjenu i pregled vozila.



Slika 1.1

Slika 6.1 - kontroleri

Slika 2.2

**Modeli**

Model je sloj u kojem se nalazi cjelokupna podatkovna logika. S obzirom da je

model podataka detaljno opisan u opisu baze podataka, on ovdje neće biti detaljno

razmatran.

Važno je za napomenuti da u RoR aplikacijama svaki model sadrži  
polje id tipa integer, polje created\_at tipa datetime, te polje updated\_at tipa datetime.

Ako nije suprotno navedeno, pretpostavlja se da svaki model sadrži navedena polja.

* **Category**

Category je model koji pohranjuje kategorije stranica.  
Sadrži polja title, slug, icon, ancestry. Polje title je polje tipa string i u njega se pohranjuje naslov kategorije, slug je polje tipa string i u njega se pohranjuje željeni URL stranice, icon je polje tipa string i u njega se pohranjuje HTML kod za prikaz ikone koja će se postaviti prije naslova kategorije, ako je to moguće. Izuzetno, ancestry je polje tipa string i služi za organizaciju podataka u stablo, te ga zahtijeva gem Ancestry.

* **Ckeditor::Asset**

Ovaj model se koristi za pokranjivanje priložaka tekstu koji su dodani kroz CKEditor.

Sadrži ime, tip, veličinu (u bajtovima), širinu i visinu (ako je slika) te uz koji unos

je asociran priložak. Ovo je apstraktni model koji je automatski generiran te za detaljni uvid u njegovu funkcionalnost se treba osvrnuti na dokumentaciju CKEditor gema.

* **Ckeditor::Attachmentfile**

Koristi se kao i Ckeditor::Asset, ali isključivo za datoteke koje nisu slike.

* **Ckeditor::Picture**

Koristi se kao i Ckeditor::Asset, ali isključivo za datoteke koje su slike.

* **Client**

Client je model koji sadrži podatke o klientima s kojima je radionica poslovala. sadrži ime i prezime klijenta u poljima nameFirst i nameLast spremljene kao string. Kontakt adresu i kontakt Telefon se pohranjuje u polja contactAdress i contactTelephone

isto kao string.

* **Image**

Image je generični model za pohranjivanje slika, svrha mu je spremanje slika korisničkih profila te od dijelova sa lagera. Ima mogućnost pohranjivanja imena slike u polje imageName kao string te također može pohraniti tip slike u polje imageType kao integer. Path je najbitnije polje u ovom model jer ono pamti gdje je na serveru slika pohranjena.

isHidden je polje koje omogućava sakrivanje slika, ako to korisnik želi.

* **Network**

Network je model za pohranjivanje socijalnih mreža koji pamti samo najosnovnije informacije poput imena, ikone i poveznice na socialnu mrežu. Ime se sprema u polje name kao string, Ikona se sprema u polje icon kao string koji sadrži HTML kod za generiranje ikone. poveznica na smau socijalnu mrežu se sprema u polje link kao string.

Polje hidden je tipa boolean i omogućuje skrivanje socijalne mreže od posjetitelja, u slučaju

da ju ne želimo prikazivati na stranici, ali ju ne želimo niti obrisati. showName je polje isto tipa boolean i omogućava prikaz imena socijalne mreže uz njezinu ikonu.

* **Order**

Model Order u sebi pohranjuje informacije o pojedinačnim radnim nalozima.

client\_id pohranjuje identifikacijski broji (integer) klijenta koji je zatražio zahvat.  
vehicle\_id pohranjuje identifikacijski broj (integer) vezila koje je klijent prijavio za zahvat.

user\_id pohranjuje identifikacijski broj (integer) referenta zaduženog za radni nalog.

Polje description pohranjuje opis željenog zahvata kao formatirani string (niz znakova).

Polje status bilježu u kojem je stanju radni nalog ( otvoren, storniran, … ) kao integer.

timeStart i timeFinish u sebi pohranjuju vrijeme i datum početka i završetka radnog naloga.

* **OrderPart**

OrderParts je pomoćni model koji služi kao posebna vrsta join tablea u to-many odnosu.  
U RoR aplikacijama je moguće napraviti join table bez modela, ali ako se žele pohranjivati

dodatni podaci u join table onda se za njega mora stvoriti model, ali ne i kontroler.

Pošto moramo pamtiti koliko je dijelova bilo potrošeno u nekom radnom nalogu treba nam

ovaj pomoćni model kako bi smo to mogli pohranjivati u sustav.  
order\_id sadrži identifikacijski broj radnog naloga, a part\_id identifikacijski broj dijela.  
Polje quantity sadrži količinu potrošenog djela. Za tip ovog polja smo se odlučili za float

radi omogućavanja lakšeg proširenja, odnosno ako se korisnik odluči unijeti količinu tekućine sustav mu to omogućava bez velikih preinaka.

* **Page**

Page model pohranjuje pojedinačne webstranice. Naslov stranice pohranjuje se u polje title kao string. URL stranice u polje slug kao string. Ikonu u polje icon kao HTML string, sadržaj u polje content kao HTML formatirani string. contentPreview je polje za potencijalna proširenja sustava (predviđeno je za dodavanje kratkog opisa stranice kako se korisniku nebi morala učitavati cijela stranice već samo njezin opis ili primjer). isPublic je boolean koji odrađuje dali neregistrirani korisnik može pristupiti ovoj stranici dok isHidden određuje dali se stranica prikazuje u popisu stranica. hasPreview je boolean koji odrađuje dali da se u popisu stranica koristi kratki opis ili puna stranica, a category\_id pohranjuje kategoriju kojoj stranica pripada. Ancestry je polje koje zahtjeva gem Ancestry i njegova je svrha da organizira stranice u stabla, međutim postoji mogućnost da će ova funkcionalnost biti izbaćena u novijim verzijama.

* **Part**

Part je model koji pohranjuje pojedinačne dijelove. Polje name sadrži ime dijela kao string,

Quantity sadrži količinu dijelova istog tipa kao cijeli broj, međutim u novijim verzijama će to biti promijenjeno u tip float kako bi se mogle u sustavu prikazivati tekućine I slični materijali.

Polje price sadrži jediničnu cijenu pojedinog dijela.

* **PartProcedure**

PartProcedure je pomoćni model koji služi kao join table. Svrha njegovog postojanja je da pohranjuje količinu korištenih dijelova nekog tipa u određenom postupku. part\_id i procedure\_id pohranjuju identifikatore djela odnosno postupka, a quantity pohranjuje količinu utrošenog dijela u postupku, kao float.

* **Procedure**

Procedure je model koji pohranjuje postupke. Ime postupka pohranjuje se u polje name, a opis postupka se pohranjuje u polje description kao string, odnosno HTML string.

minNumOfWorkers je polje koje pohranjuje minimalan broj zaposlenika potrebnih da se postupak obavi. Polje duration pohranjuje trajanje samog postupka u satima, a

manHour je polje koje pohranjuje cijenu čovjek sata za ovaj postupak.

* **Role**

Role je model koji pohranjuje uloge korisnika u sustavu te je jedan od najbitnijih modela jer određuje koji korisnik ima kakva prava pristupa u programu. Sadrži polje name u koje se pohranjuje ime uloge kao string i dva booleana za svaki kontroler u aplikaciji koji određuju dali korisnik može dodavati, izmijenjivati i brisati u tom kontroleru, odnosno dali smije gledati sadržaj kojim raspolaže taj kontroler.

* **Setting**

Setting je model koji pohranjuje postavke stranice, sadrži ime stranice u polju pageTitle, kao string, opis stranice u polju pageDescription kao formatirani string te identifikatore slika koje će se koristiti za favikonu stranice, odnosno za sliku na socijalnim mrežama, pohranjene u poljima pageFavicon i pagePreview.

* **Statement**

Statement je model koji sadrži dnevne, mjesečne i godišnje izračune. U polje type spremamo vrstu izračuna, 0 je za dnevni, 1 za tjedni, 2 za mjesečni i 3 za godišnji izračun što je bitno radi optimizacije izrade izračuna i optimizacije pretraživanja izračuna. statementDate pohranjuje datum izrade izračuna, a polje money sadrži izračunatu bilansu.

* **Vehicle**

Model Vehicle sadrži vozila klienata, vehicleModel polje pohranjuje marku automobila kao string, godina izgradnje vozila je zapisana u polju buildYear kao integer i

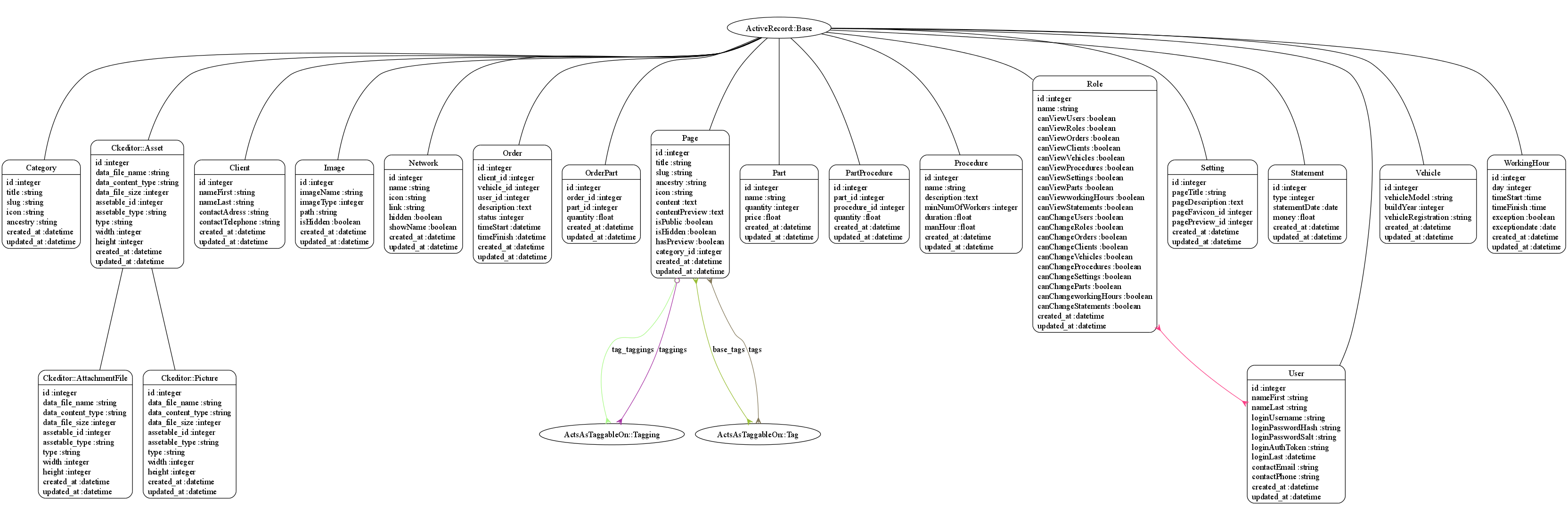
registracija vozila je zapisana u polju vehicleRegistration kao string.

* **WorkingHour**

WorkingHours je model koji sadrži radno vrijeme radionice.. Model je izrazito bitan za izradu izračuna jer se oni moraju raditi na kraju radnog vremena, koje je određeno ovim modelom. Polje day sadrži dan u tjednu na koji se radno vrijeme odnosi, 0 za ponedjeljak, 1 za utorak itd. dok timeStart i timeFinish su polja u koja se pohranjuje vrijeme otvaranja i zatvaranja radionice. Polje exception određuje dali je unos izvanredan te se neće rotirati kao ostala radna vremena u sustavu već će se pojaviti samo na datum iznimke i nikada više ( ovo je potrebno za praznike i posebne okolnosti ). Polje exceptiondate pohranjuje datum iznimke.

* **User**

User je definitivno najvažniji model u sustavu, on sadrži sve podatke potrebne za funkcioniranje korisnika u sustavu i omogućava postojanje korisnika. Ime i prezime korisnika su pohranjeni u poljima nameFirst i nameLast kao string dok se korisničko ime korisnika se pohranjuje u polje loginUsername kao string. Korisnikova lozinka se pomoću bycript gema razlaže u njezinu hash i salt komponentu koje se pohranjuju u loginHash i loginSalt polja ovog modela kao stringovi. loginAuthToken je polje koje sadrži nasumično generirani unikatni autentifikacijski token, token ima dvije svrhe, za jednoznačnu identifikaciju korisnika iz session kolačića sa korisnikovog pretraživača, te za mogućnost potencijalnog proširenja sustava i stvaranje eksternog APIja za komunikaciju sa aplikacijom. Polje loginLast pohranjuje kada se korisnik zadnji puta prijavio u sustav, a polja contactEmail i contactPhone sadrže email adresu i telefonski broji korisnika.



Slika 6.2 - modeli