Invoicing App – README

**Sadržaj :**

1. DB – InvoicingApp\_Nail.bak
2. InvoicingApp\_Nail.sln
   1. InvoicingApp.DAL – Data Access Layer
   2. InvoicingApp.Web – MVC5 , Razor engine
   3. InvoicingApp\_Nail – SQL DB Project

**Walktrough:**

**Baza podataka** je napravljena Code-First pristupom i sve referentne klase kao i migracijska konfiguracija i detalji se nalaze u InvoicingApp.DAL projektu.

Prilikom izrade Entitiya, mislio sam se da li da stavim unique key string vrijednosti koje su imale zvjezdice u excel dokumentu, ali sam se ipak odlučio da stavim integer identifikatore sa automatskom inkrementacijom kao primarne ključeve za sve tabele – radi best practice-a i olakšanog rada kasnije, a unikantnost podataka sam postigao kreiranje indexa kroz code first approach, umjesto kompozitnog ključa na Invoice tabeli sada imamo kompozitni index koji pravi istu server side validaciju koju bi postigli i sa kompozitnim ključem.

**InvoicingApp.Web** sadrži glavninu posla, nisam imao dovoljno vremena da frontend „ušminkam“, ali sam se potrudio da ono što se nalazi na site-u i radi.

Na vrhu aplikacije se nalazi navigacijski menu sa dvije opcije „Clients“ i „Invoices“

***Clients*** – na vrhu se nalazi link za kreiranje novog zapisa „Create New“, ono što je bitno apomenuti za kreiranje i editovanje da je implementiran i client side validacija gdje je CompanyName unikatan, naravno postoji i server side valdiacija. Iznad grida se nalazi textbox za filtriranje na principu „Contains“, koji će pokazati sv epodatke ukoliko se izbriše filter i uradi ponovni post-back. Unutar grid je implementiran sort na CompanyName, koji na klik mijenja asc u desc i obratno sortiranje. Implementiran je i paging sa pagesize-om od 5. CRUD operacije sve implementirane.

***Invoices*** - na vrhu se nalazi link za kreiranje novog zapisa „Create New“, ono što je bitno apomenuti za kreiranje i editovanje da je implementiran i client side validacija gdje je kombinacija InvoiceNumbera i CompanyId unikatan, naravno postoji i server side valdiacija. Iznad grida se nalazi textbox za filtriranje na principu „Contains“ po CompanyName i InvoiceNumber, koji će pokazati sve podatke ukoliko se izbriše filter i uradi ponovni post-back. Unutar grid je implementiran sort na InvoiceNumber i CompanyName, koji na klik mijenja asc u desc i obratno sortiranje. Implementiran je i paging sa pagesize-om od 5. CRUD operacije sve implementirane. Sa desne strane se nalazi link sa natpisom „Run Report“ koji će prikazati korisniku InvoicePrint report u pdf formatu u novom tabu i time omogućiti printanje istog, downlodovanje i slično, report uzima parametar clientId na osnovu kojeg vraća podatke. Ispod grida se nalaze dva datepickera koji su neophodni da bi se downlodovao/printao chart, tek nakon što se i start i end date unesu tada će button pokrenuti report, u protivnom se neće ništa desiti(client side validacija jer je za report potrebno imati dva date parametra).

**Reports:**

Unutar InvoicingApp.Web projekta se nalazi folder „Reports“ gdje su sadržani rdlc reporti za invoice print i char. Pozadina ovih reporta se nalazi u bazi podataka, dodatna dva view-a i tri stored procedure.

**SQL DB Project** – Je samo ukoliko bi htjeli preko schema compare da preuzmete bazu podataka.

Ovo je bilo ukratko o rješenju, a detaljnije ćemo na prezentaciji riješenja.

Svako dobro,

-Nail