```
@startuml
skinparam componentStyle rectangle

package "ArchOp Application" {
    package "ViewModels" {


         class RegisterViewModel {
            - NavStore navStore
            - string Email
            - string Password
            - string RePass
            - Brush Color1
            - Brush Color2
            - Brush Color3
            - ICommand RegisterCommand
            - ICommand BackCommand
            --
            + RegisterViewModel(NavStore navStore)
            + Task<int> Register()
            + void RegisterButton()
            + void BackButton()
        }

        class ViewModelBase {
            + event PropertyChanged
            + bool SetProperty<T>(ref T storage, T value, [CallerMemberName]
string propertyName = null)
            + void OnPropertyChanged([CallerMemberName] string propertyName =
null)
        }

        class SentInvoicesViewModel {
            - NavStore navStore
            - ObservableCollection<Invoice> Invoices
            - ICommand DownloadInvoiceCommand
            --
            + SentInvoicesViewModel(NavStore navStore)
            + LoadInvoicesAsync()
            + DownloadInvoice(Invoice invoice)
        }

        class MainWindowViewModel {
        - NavStore navStore
        --
        + ViewModelBase CurrentViewModel
        + MainWindowViewModel()
        + MainWindowViewModel(NavStore navStore)
        }

        class AddCompanyViewModel {
            - NavStore navStore
            - string CompanyName
            - string CompanyAddress
    - ICommand AddCompanyCommand
    --
    + AddCompanyViewModel(NavStore navStore)
    + AddToUserAddedCompanies()
    + AddCompanyButton()
}

class DashboardViewModel {
    - NavStore navStore
    - ICommand NavToLoginCommand
    - ICommand NavToRegisterCommand
    --
    + DashboardViewModel(NavStore navStore)
    + ExecuteNavToRegister()
    + ExecuteNavToLogin()
}

class HomePageViewModel {
    - NavStore navStore
    - Page CurrentPage
    - ICommand NavToInvoiceMakerCommand
    - ICommand NavToSentInvoiceCommand
    - ICommand NavToSetCompanyCommand
    - ICommand NavToAddCompanyCommand
    - ICommand LogOutCommand
    --
    + HomePageViewModel(NavStore navStore)
    + ExecuteNavToInvoice()
    + ExecuteNavToSentInvoice()
    + ExecuteNavToSetCompany()
    + ExecuteNavToAddCompany()
    + ExecuteLogOut()
}

class InvoiceMakerViewModel {
    - NavStore navStore
    - string SummaryDescription
    - DateTime DueDate
    - ObservableCollection<Item> InvoiceItems
    - string Name
    - string Description
    - string Quantity
    - string Price
    - string[] AvailableVat
    - double? SelectedVat
    - ObservableCollection<Company?> UserCompanies
    - Company SelectedCompany
    - Company OwnCompany
    - bool IsCreateInvoiceEnabled
    --
    + InvoiceMakerViewModel(NavStore navStore)
    + Task<bool> CreateInvoice()
    + Task<byte[]> CreatePDF(int invoiceId, int invoiceUserId)
    + void AddItem()
```

```
        + Task LoadUserCompaniesAsync()
        + Task LoadOwnCompanyAsync()
    }

    class LoginViewModel {
        - NavStore navStore
        - string Email
        - string Password
        - bool IsLoginButtonEnabled
        --
        + LoginViewModel(NavStore navStore)
        + Task<bool> Login(string password)
        + void BackButton()
        + void LoginButton()
    }
}

package "Models" {
    class Company {
        + int CompanyId
        + string CompanyName
        + string CompanyAddress
    }

    class Invoice {
        + int InvoiceId
        + string UserId
        + string? InvoiceYear
        + int? InvoiceUserId
        + string InvoiceDisplayName
    }

    class Users {
        + int UserRowId
        + string SupabaseUserId
        + int? UserCompanyId
        + string? UserAddedCompaniesId
    }

    class Item {
        + string Name
        + string? Description
        + double Price
        + double Quantity
        + double TotalPrice
        + double? Vat
        --
        + double DisplayVat
        + double ItemBrutto
        + double ItemNetto
        + double Amount()
        + double Brutto()
        + double Netto()
    }
```

```
        class DBRequests {
            + static Task<List<Invoice>> GetInvoicesAsync(string userId)
            + static Task<bool> AlreadyAddedToUserCompanies(string companyName)
            + static Task<Company> GetCompanyByName(string companyName)
            + static Task<Company> AddCompany(string companyName, string
companyAddress)
            + static Task AddToUserAddedCompanies(string companyId)
            + static Task<int> GetInvoiceUserId()
            + static Task<int> GetNewInvoiceId()
            + static Task<int> GetUserInvoiceId()
            + static Task<List<Company>> GetUserAddedCompaniesAsync()
            + static Task<Company> GetOwnCompany()
            + static Task InsertInvoice(byte[] pdfData, string pdfPath, string
invoiceId, string userId, string invoiceYear, int invoiceUserId)
        }
    }

    package "Views" {

        class SentInvoicesWindow {
            + ListView Invoices
            + Button Download
        }
        class MainWindow {
        + Frame MainFrame
        }

        class AddCompanyPage {
            + TextBox CompanyName
            + TextBox CompanyAddress
            + Button AddCompany
        }

        class DashboardPage {
            + Button Login
            + Button Register
        }

        class HomePage {
            + Button InvoiceMaker
            + Button SentInvoices
            + Button SetCompany
            + Button AddCompany
            + Button Logout
        }
        class RegisterPage {
            + TextBox Email
            + PasswordBox Password
            + PasswordBox RePass
            + Button Register
            + Button Back
        }
    }
    package "Services" {
        class App {
            + SupabaseClient SupabaseClient
        }

        class NavStore {
            + event Action CurrentViewModelChanged
            + ViewModelBase CurrentViewModel
            + static MainWindowViewModel MainWindowViewModel
            --
            + void OnCurrentViewModelChanged()
        }
    }

    package "Storage" {
        class SupabaseClient {
            + Task<byte[]> DownloadPublicFile(string filePath)
        }
    }
}

' Relationships
ViewModelBase <|-- RegisterViewModel : "Inherits"
ViewModelBase <|-- SentInvoicesViewModel : "Inherits"
ViewModelBase <|-- MainWindowViewModel : "Inherits"
ViewModelBase <|-- AddCompanyViewModel : "Inherits"
ViewModelBase <|-- DashboardViewModel : "Inherits"
ViewModelBase <|-- HomePageViewModel : "Inherits"
ViewModelBase <|-- InvoiceMakerViewModel : "Inherits"
ViewModelBase <|-- LoginViewModel : "Inherits"
MainWindowViewModel --> NavStore : "Observes current state"
MainWindowViewModel --> ViewModelBase : "Returns current"
MainWindow --> MainWindowViewModel : "Binds to"
SentInvoicesViewModel --> Invoice : "Manages"
SentInvoicesViewModel --> DBRequests : "Calls"
SentInvoicesViewModel --> App : "Uses"
SentInvoicesViewModel --> NavStore : "Depends on"
SentInvoicesWindow --> SentInvoicesViewModel : "Binds to"
DBRequests --> App : "Uses"
DBRequests --> Invoice : "Returns"
App --> SupabaseClient : "Accesses"
SupabaseClient --> "Invoices Storage Bucket" : "Downloads from"
Invoice --> Company : "References (InvoiceUserId)"
Invoice --> Item : "Contains multiple"
Users --> Invoice : "Owns"
Users --> Company : "References (UserCompanyId)"
Users --> Company : "Can Add (UserAddedCompaniesId)"
AddCompanyViewModel --> Company : "Adds"
AddCompanyViewModel --> DBRequests : "Calls"
AddCompanyViewModel --> NavStore : "Navigates with"
DashboardViewModel --> NavStore : "Navigates with"
DashboardViewModel --> HomePageViewModel : "Navigates to"
HomePageViewModel --> AddCompanyViewModel : "Navigates to"
HomePageViewModel --> SentInvoicesViewModel : "Navigates to"
HomePageViewModel --> Invoice : "Creates"
HomePageViewModel --> NavStore : "Uses"
NavStore --> ViewModelBase : "Uses"
NavStore --> MainWindowViewModel : "Holds instance"
InvoiceMakerViewModel --> Item : "Manages"
InvoiceMakerViewModel --> DBRequests : "Calls"
InvoiceMakerViewModel --> NavStore : "Uses"
InvoiceMakerViewModel --> Company : "Manages"
InvoiceMakerViewModel --> Invoice : "Creates"
LoginViewModel --> NavStore : "Uses"
LoginViewModel --> DashboardViewModel : "Navigates to"
LoginViewModel --> HomePageViewModel : "Navigates to"
RegisterViewModel --> NavStore : "Uses"
RegisterViewModel --> App : "Uses SupabaseClient"
RegisterViewModel --> RegexUtilities : "Validates inputs"
RegisterViewModel --> DashboardViewModel : "Navigates to"
RegisterViewModel --> HomePageViewModel : "Navigates to"
RegisterPage --> RegisterViewModel : "Binds to"
@enduml
```