Build Instructions

Source: https://github.com/paragon-it-solutions/firefish_web_api

PREREQUESTIES:

- ❖ MS Visual Studio or JetBrains Rider
- ❖ .NET 8 / 9 SDK
- ❖ MS SQL Server
- ❖ Node.js
- ❖ NPM

Build Instructions .NET

1. Open Firefish.sln file in visual studio / rider
2. Run clean all and rebuild
3. Restore nuget packages
4. Change ConnectionString in SqlConnectionHelper.cs to match your local copy of provided .BAK
5. Run with F5
6. Browse Swagger at localhost:{YOURAPPPORT}/swagger for api and model documentation

Build Instructions React

1. Go to \Firefish.API\react-app\
2. Restore npm packages
3. Go to \Firefish.API\react-app\src\components\shared and update ApiConfig.js to URL of running dotnet app
4. Run command “npm run build” in FIrefish.API\react-app\firefish directory
5. Run command “serve -s build”
6. Browse running site

Backlog and Bugs

.NET App Backlog

Implement authentication and authorization for production using JWT
Add pagination for get all candidates method
Rework candidates to return a list of skills with their details rather than having to do a get request for each individual candidate's skills
Add more comprehensive unit tests and tests edge cases / erroneous cases
Implement logging with better error handling
Add https for live with cert
Standardize how things are done in code for production – just wanted to show variety of ways to do things, but real app should follow standardize approaches

SQL Backlog

Possibly consider using DBSPROC instead of raw SQL in .net and call SPROC from C#
Consider adding indexes to foreign keys (e,g CandidateId or SkillId in CandidateSkill table) as these are often used on joins.
Maybe change to using identity on tables

React App Known Bugs

Issue with selectbox on add skill – value not show in box but updates OK
Issue with display of skills for candidates - index is out by -1 so skills shown on candidate 2 are actually for candidate 1 etc

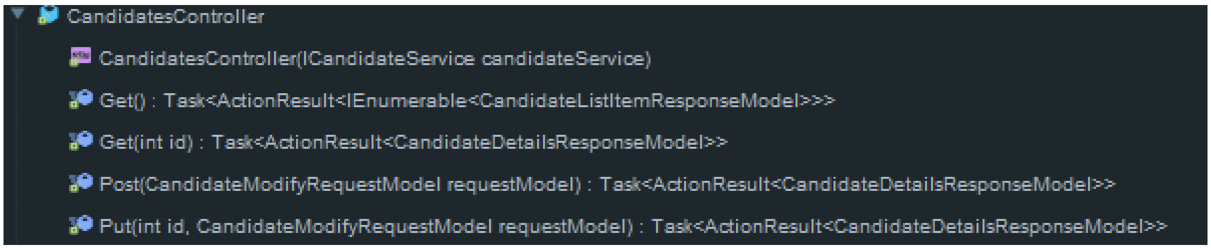
React Backlog

Standardize style across all pages
Implement unit and integration testing
Standardize error returns and handling (some are 'snackbar' some are 'modal')
Refactor components out into individual classes and separate styling
Use styled to use CSS to style rather than using sx style on React
Add validation to each field for submissions
Utilize errors returned from API to provide more specific feedback to end user

Diagrams

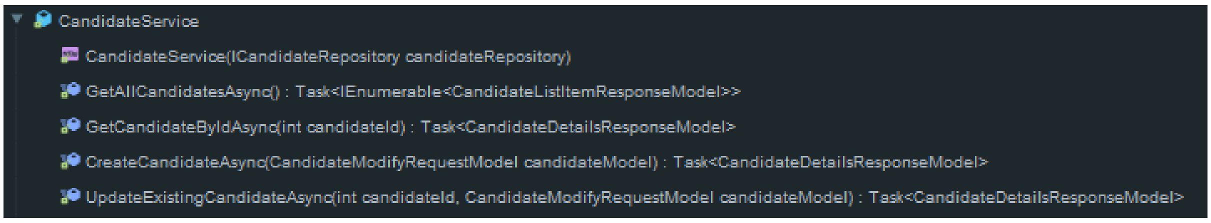
Key Design Diagrams

Repository Pattern and Service Layer -> Mapped to API Controller



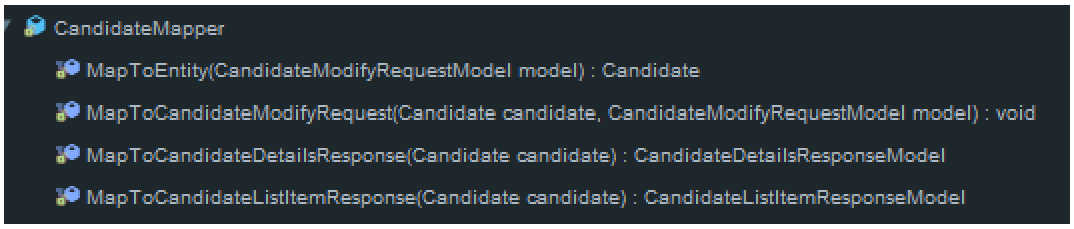
```
CandidatesController
    CandidatesController(ICandidateService candidateService)
    Get() : Task<ActionResult<IEnumerable<CandidateListItemResponseModel>>>
    Get(int id) : Task<ActionResult<CandidateDetailsResponseModel>>
    Post(CandidateModifyRequestModel requestModel) : Task<ActionResult<CandidateDetailsResponseModel>>
    Put(int id, CandidateModifyRequestModel requestModel) : Task<ActionResult<CandidateDetailsResponseModel>>
```

CandidatesController requests mapped data from CandidateService



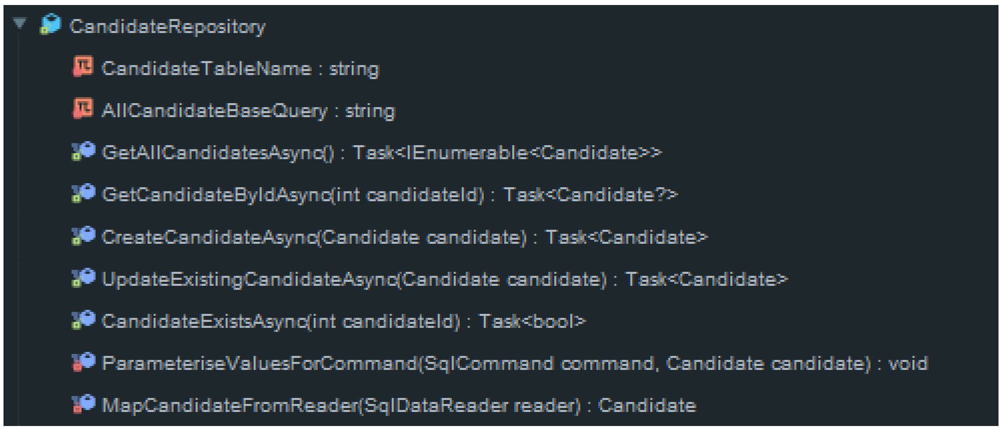
```
CandidateService
    CandidateService(ICandidateRepository candidateRepository)
    GetAllCandidatesAsync() : Task<IEnumerable<CandidateListItemResponseModel>>
    GetCandidateByIdAsync(int candidateId) : Task<CandidateDetailsResponseModel>
    CreateCandidateAsync(CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
    UpdateExistingCandidateAsync(int candidateId, CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
```

CandidateService uses CandidateMapper to map entities to models FROM Repository

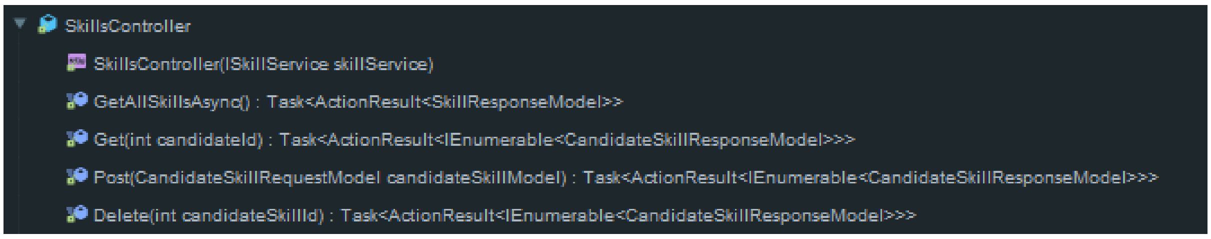


```
CandidateMapper
    MapToEntity(CandidateModifyRequestModel model) : Candidate
    MapToCandidateModifyRequest(Candidate candidate, CandidateModifyRequestModel model) : void
    MapToCandidateDetailsResponse(Candidate candidate) : CandidateDetailsResponseModel
    MapToCandidateListItemResponse(Candidate candidate) : CandidateListItemResponseModel
```

CandidatesRepository requests raw data from SQL Server directly

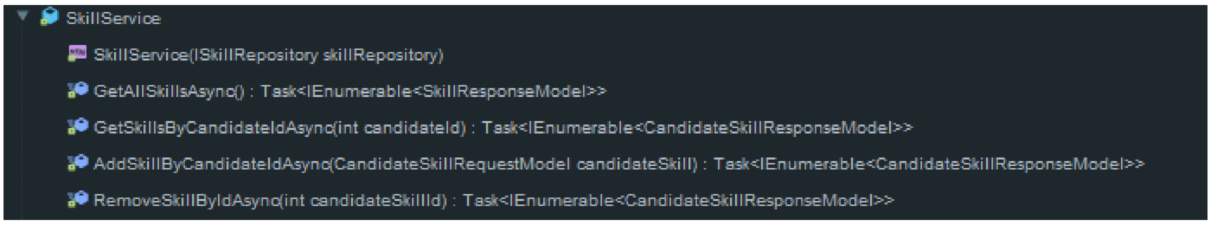


```
CandidateRepository
    CandidateTableName : string
    AllCandidateBaseQuery : string
    GetAllCandidatesAsync() : Task<IEnumerable<Candidate>>
    GetCandidateByIdAsync(int candidateId) : Task<Candidate?>
    CreateCandidateAsync(Candidate candidate) : Task<Candidate>
    UpdateExistingCandidateAsync(Candidate candidate) : Task<Candidate>
    CandidateExistsAsync(int candidateId) : Task<bool>
    ParameteriseValuesForCommand(SqlCommand command, Candidate candidate) : void
    MapCandidateFromReader(SqlDataReader reader) : Candidate
```



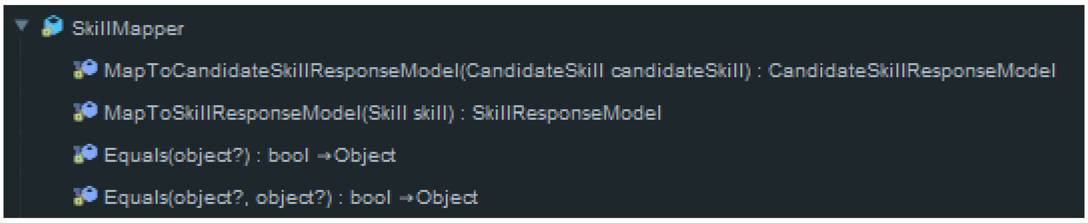
```
SkillsController
    SkillsController(ISkillService skillService)
    GetAllSkillsAsync() : Task<ActionResult<SkillResponseModel>>
    Get(int candidateId) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
    Post(CandidateSkillRequestModel candidateSkillModel) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
    Delete(int candidateSkillId) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
```

SkillsController requests mapped data from SkillService



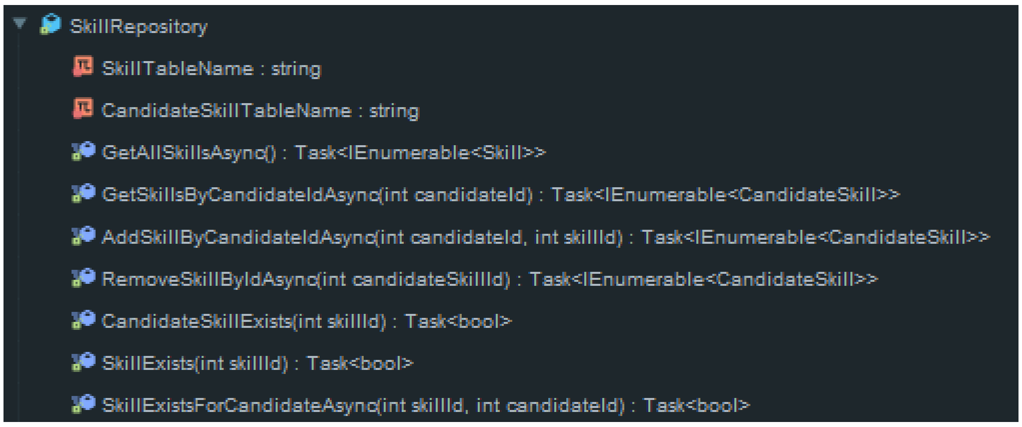
```
SkillService
    SkillService(ISkillRepository skillRepository)
    GetAllSkillsAsync() : Task<IEnumerable<SkillResponseModel>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkillResponseModel>>
    AddSkillByCandidateIdAsync(CandidateSkillRequestModel candidateSkill) : Task<IEnumerable<CandidateSkillResponseModel>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkillResponseModel>>
```

SkillService uses SkillMapper to map entities to models FROM Repository



```
SkillMapper
    MapToCandidateSkillResponseModel(CandidateSkill candidateSkill) : CandidateSkillResponseModel
    MapToSkillResponseModel(Skill skill) : SkillResponseModel
    Equals(object?) : bool →Object
    Equals(object?, object?) : bool →Object
```

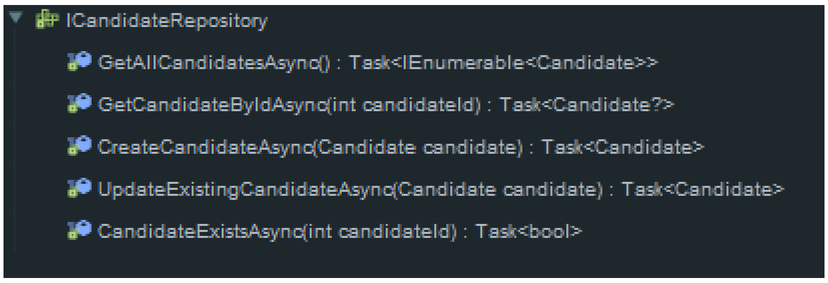
SkillRepository requests raw data from SQL Server directly



```
SkillRepository
    SkillTableName : string
    CandidateSkillTableName : string
    GetAllSkillsAsync() : Task<IEnumerable<Skill>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkill>>
    AddSkillByCandidateIdAsync(int candidateId, int skillId) : Task<IEnumerable<CandidateSkill>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkill>>
    CandidateSkillExists(int skillId) : Task<bool>
    SkillExists(int skillId) : Task<bool>
    SkillExistsForCandidateAsync(int skillId, int candidateId) : Task<bool>
```

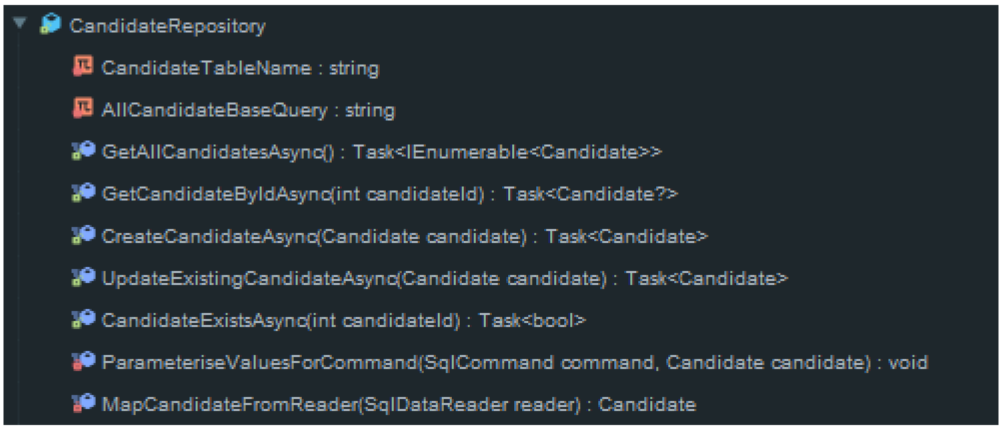
Interface Implementation

Repositories

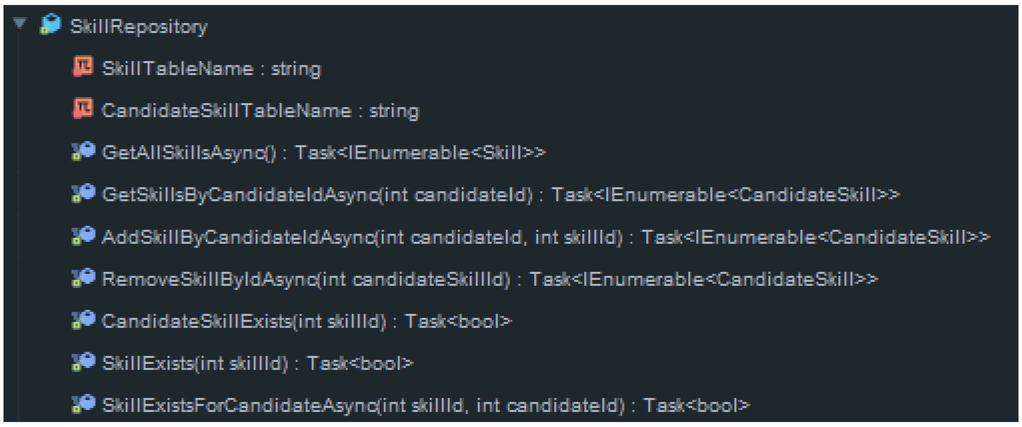


```
ICandidateRepository
    GetAllCandidatesAsync() : Task<IEnumerable<Candidate>>
    GetCandidateByIdAsync(int candidateId) : Task<Candidate?>
    CreateCandidateAsync(Candidate candidate) : Task<Candidate>
    UpdateExistingCandidateAsync(Candidate candidate) : Task<Candidate>
    CandidateExistsAsync(int candidateId) : Task<bool>
```

CandidateRepository implements ICandidateRepository

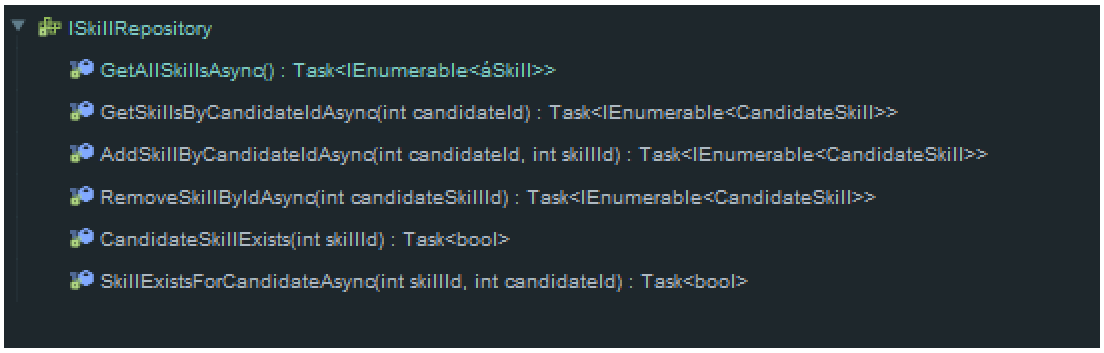


```
CandidateRepository
    CandidateTableName : string
    AllCandidateBaseQuery : string
    GetAllCandidatesAsync() : Task<IEnumerable<Candidate>>
    GetCandidateByIdAsync(int candidateId) : Task<Candidate?>
    CreateCandidateAsync(Candidate candidate) : Task<Candidate>
    UpdateExistingCandidateAsync(Candidate candidate) : Task<Candidate>
    CandidateExistsAsync(int candidateId) : Task<bool>
    ParameteriseValuesForCommand(SqlCommand command, Candidate candidate) : void
    MapCandidateFromReader(SqlDataReader reader) : Candidate
```



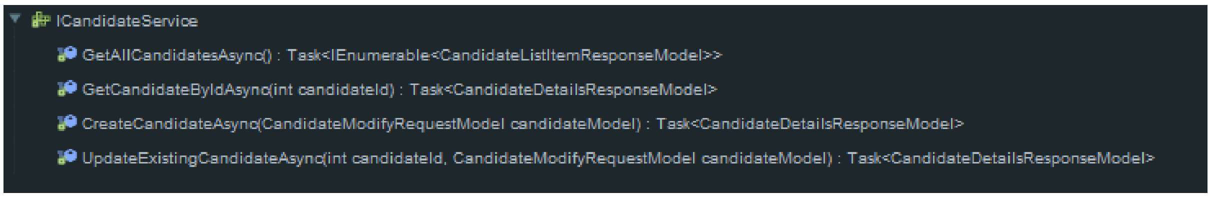
```
SkillRepository
    SkillTableName : string
    CandidateSkillTableName : string
    GetAllSkillsAsync() : Task<IEnumerable<Skill>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkill>>
    AddSkillByCandidateIdAsync(int candidateId, int skillId) : Task<IEnumerable<CandidateSkill>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkill>>
    CandidateSkillExists(int skillId) : Task<bool>
    SkillExists(int skillId) : Task<bool>
    SkillExistsForCandidateAsync(int skillId, int candidateId) : Task<bool>
```

CandidateRepository implements ISkillRepository



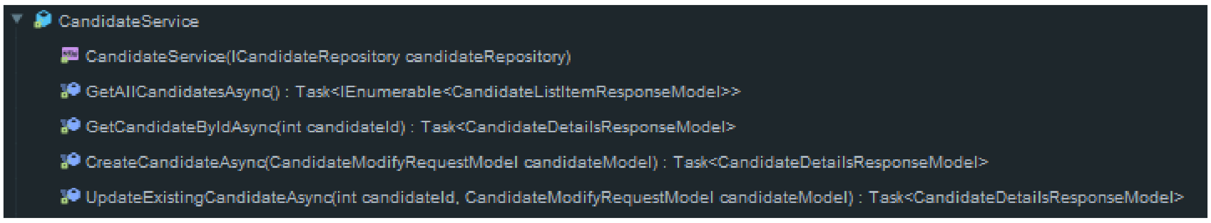
```
ISkillRepository
    GetAllSkillsAsync() : Task<IEnumerable<Skill>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkill>>
    AddSkillByCandidateIdAsync(int candidateId, int skillId) : Task<IEnumerable<CandidateSkill>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkill>>
    CandidateSkillExists(int skillId) : Task<bool>
    SkillExistsForCandidateAsync(int skillId, int candidateId) : Task<bool>
```

Services

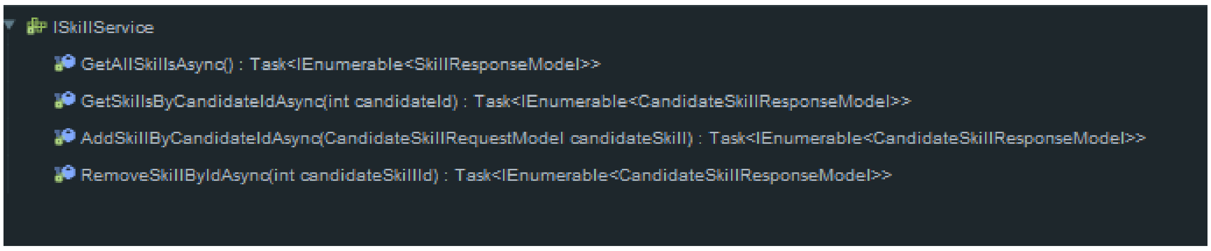


```
ICandidateService
    GetAllCandidatesAsync() : Task<IEnumerable<CandidateListItemResponseModel>>
    GetCandidateByIdAsync(int candidateId) : Task<CandidateDetailsResponseModel>
    CreateCandidateAsync(CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
    UpdateExistingCandidateAsync(int candidateId, CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
```

CandidateService Implements ICandidateService

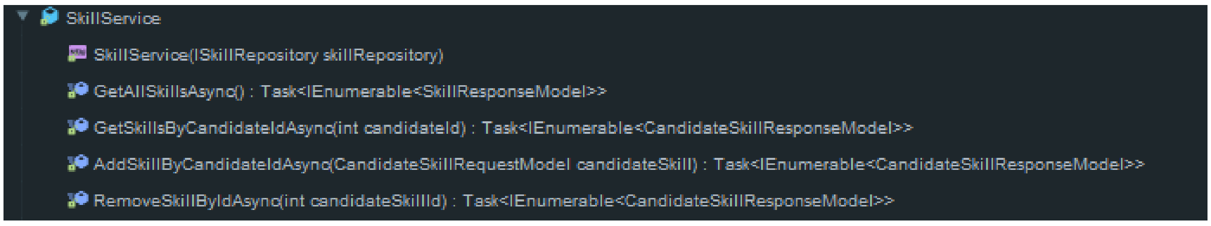


```
CandidateService
    CandidateService(ICandidateRepository candidateRepository)
    GetAllCandidatesAsync() : Task<IEnumerable<CandidateListItemResponseModel>>
    GetCandidateByIdAsync(int candidateId) : Task<CandidateDetailsResponseModel>
    CreateCandidateAsync(CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
    UpdateExistingCandidateAsync(int candidateId, CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
```



```
ISkillService
    GetAllSkillsAsync() : Task<IEnumerable<SkillResponseModel>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkillResponseModel>>
    AddSkillByCandidateIdAsync(CandidateSkillRequestModel candidateSkill) : Task<IEnumerable<CandidateSkillResponseModel>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkillResponseModel>>
```

SkillService implements ISkillService

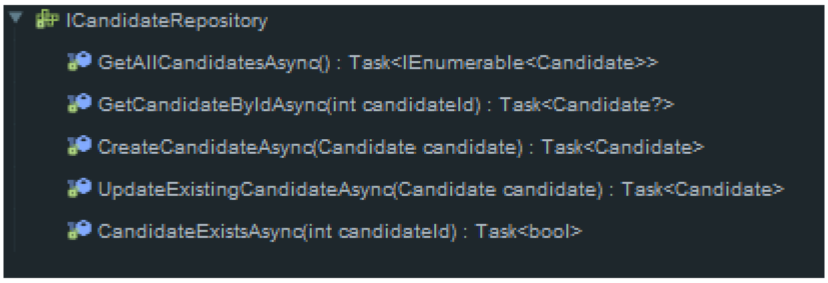


```
SkillService
    SkillService(ISkillRepository skillRepository)
    GetAllSkillsAsync() : Task<IEnumerable<SkillResponseModel>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkillResponseModel>>
    AddSkillByCandidateIdAsync(CandidateSkillRequestModel candidateSkill) : Task<IEnumerable<CandidateSkillResponseModel>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkillResponseModel>>
```

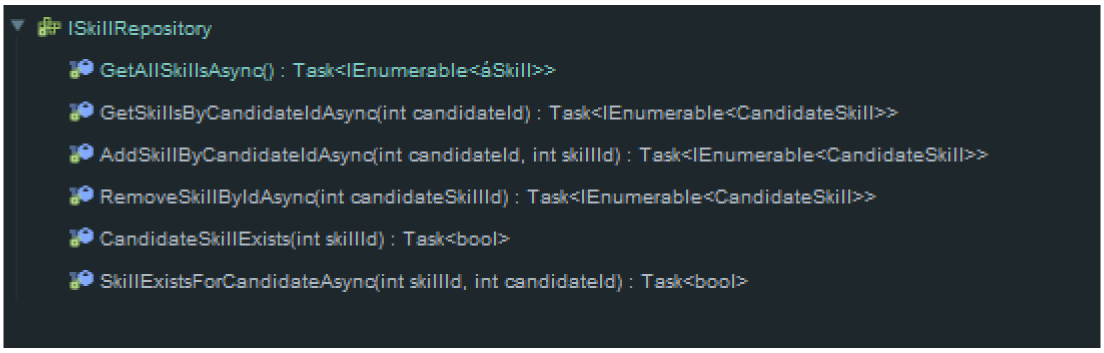
Standalone Class Diagrams

Firefish.Core

Firefish.Core.Contracts.Repositories

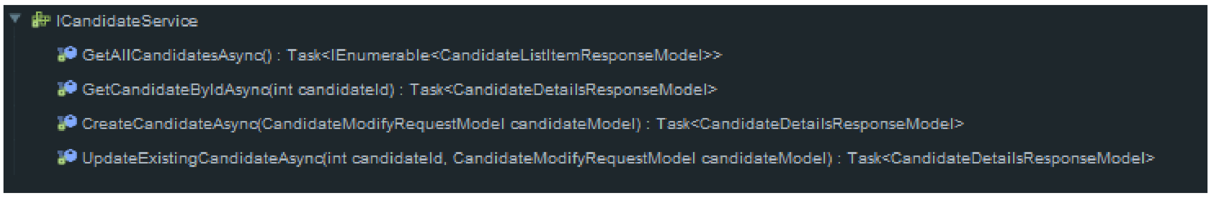


```
ICandidateRepository
    GetAllCandidatesAsync() : Task<IEnumerable<Candidate>>
    GetCandidateByIdAsync(int candidateId) : Task<Candidate?>
    CreateCandidateAsync(Candidate candidate) : Task<Candidate>
    UpdateExistingCandidateAsync(Candidate candidate) : Task<Candidate>
    CandidateExistsAsync(int candidateId) : Task<bool>
```

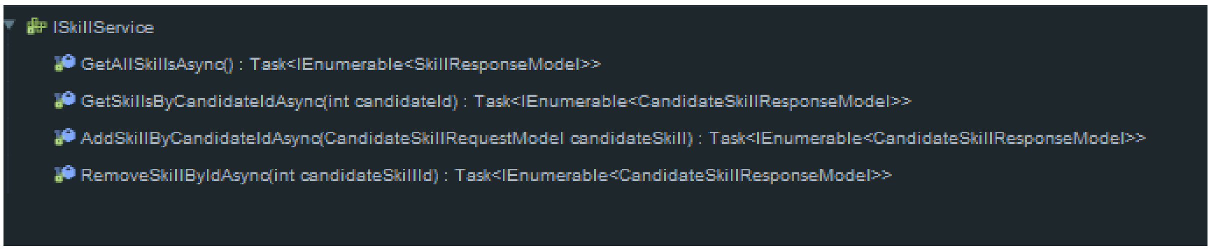


```
ISkillRepository
    GetAllSkillsAsync() : Task<IEnumerable<áSkill>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkill>>
    AddSkillByCandidateIdAsync(int candidateId, int skillId) : Task<IEnumerable<CandidateSkill>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkill>>
    CandidateSkillExists(int skillId) : Task<bool>
    SkillExistsForCandidateAsync(int skillId, int candidateId) : Task<bool>
```

Firefish.Core.Contracts.Services

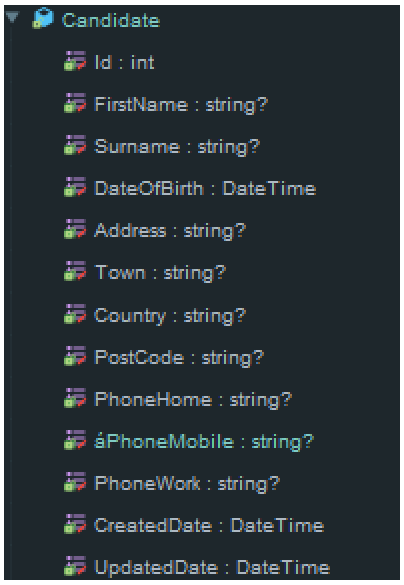


```
ICandidateService
    GetAllCandidatesAsync() : Task<IEnumerable<CandidateListItemResponseModel>>
    GetCandidateByIdAsync(int candidateId) : Task<CandidateDetailsResponseModel>
    CreateCandidateAsync(CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
    UpdateExistingCandidateAsync(int candidateId, CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
```

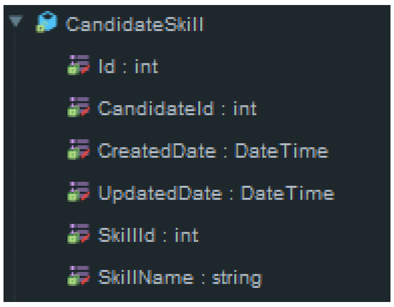


```
ISkillService
    GetAllSkillsAsync() : Task<IEnumerable<SkillResponseModel>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkillResponseModel>>
    AddSkillByCandidateIdAsync(CandidateSkillRequestModel candidateSkill) : Task<IEnumerable<CandidateSkillResponseModel>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkillResponseModel>>
```

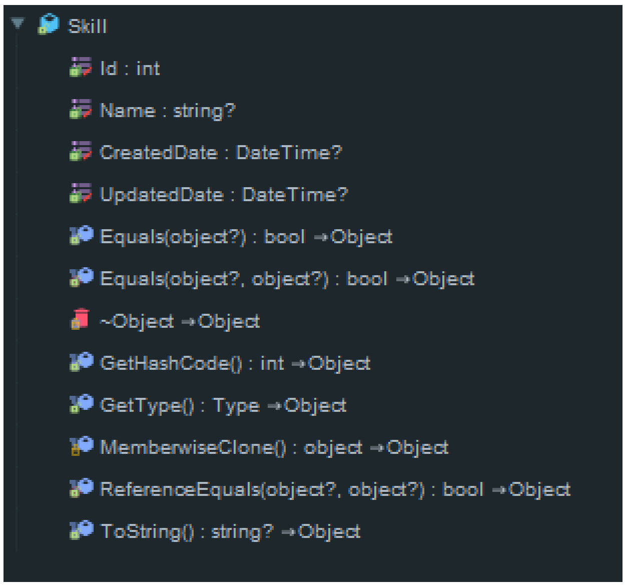
Firefish.Core.Entites



```
Candidate
    Id : int
    FirstName : string?
    Surname : string?
    DateOfBirth : DateTime
    Address : string?
    Town : string?
    Country : string?
    PostCode : string?
    PhoneHome : string?
    áPhoneMobile : string?
    PhoneWork : string?
    CreatedDate : DateTime
    UpdatedDate : DateTime
```

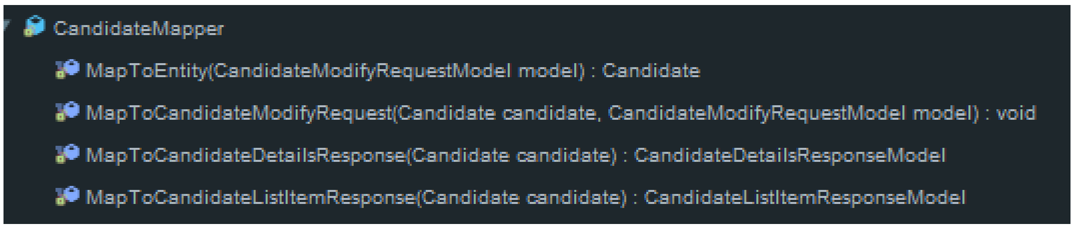


```
CandidateSkill
    Id : int
    CandidateId : int
    CreatedDate : DateTime
    UpdatedDate : DateTime
    SkillId : int
    SkillName : string
```

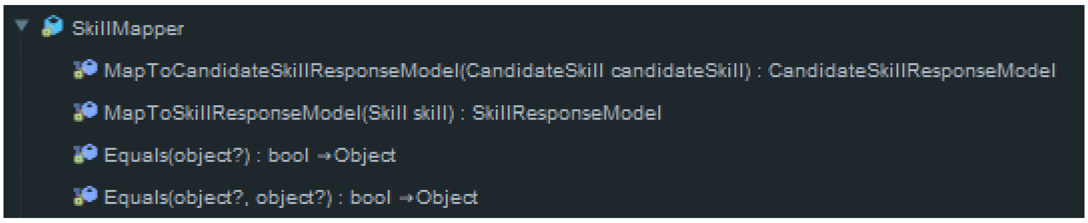


```
Skill
    Id : int
    Name : string?
    CreatedDate : DateTime?
    UpdatedDate : DateTime?
    Equals(object?) : bool →Object
    Equals(object?, object?) : bool →Object
    ~Object →Object
    GetHashCode() : int →Object
    GetType() : Type →Object
    MemberwiseClone() : object →Object
    ReferenceEquals(object?, object?) : bool →Object
    ToString() : string? →Object
```

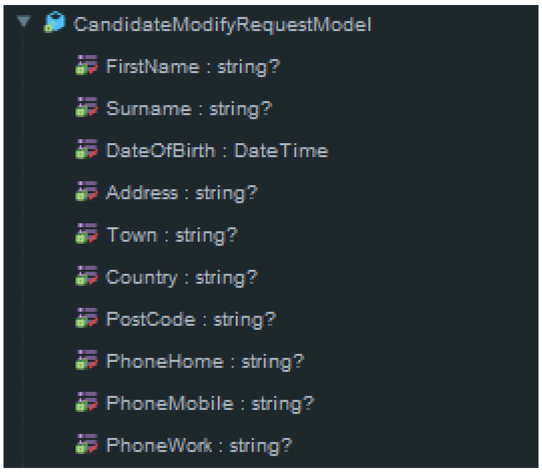
Firefish.Core.Mappers



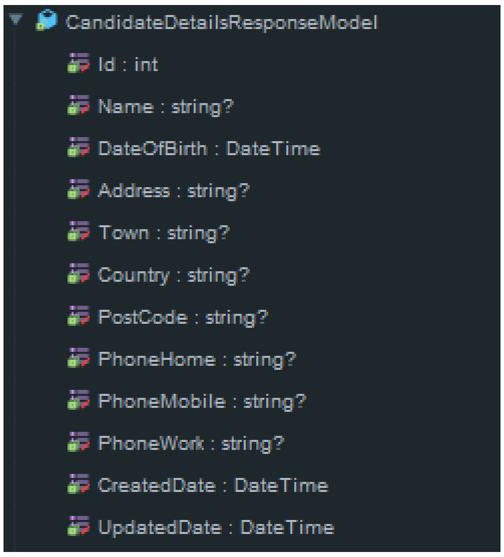
```
CandidateMapper
    MapToEntity(CandidateModifyRequestModel model) : Candidate
    MapToCandidateModifyRequest(Candidate candidate, CandidateModifyRequestModel model) : void
    MapToCandidateDetailsResponse(Candidate candidate) : CandidateDetailsResponseModel
    MapToCandidateListItemResponse(Candidate candidate) : CandidateListItemResponseModel
```

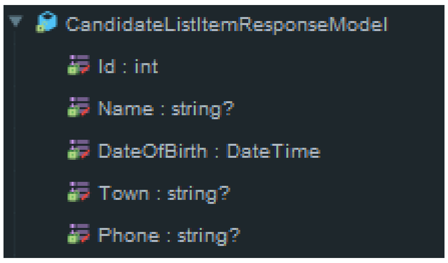

```
SkillMapper
    MapToCandidateSkillResponseModel(CandidateSkill candidateSkill) : CandidateSkillResponseModel
    MapToSkillResponseModel(Skill skill) : SkillResponseModel
    Equals(object?) : bool →Object
    Equals(object?, object?) : bool →Object
```

Firefish.Core.Models.Candidate.Requests

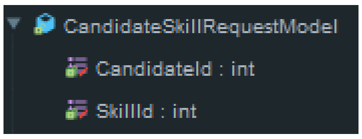

```
CandidateModifyRequestModel
    FirstName : string?
    Surname : string?
    DateOfBirth : DateTime
    Address : string?
    Town : string?
    Country : string?
    PostCode : string?
    PhoneHome : string?
    PhoneMobile : string?
    PhoneWork : string?
```

Firefish.Core.Models.Candidate.Responses

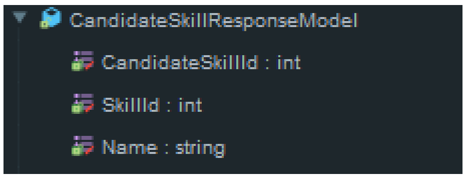

```
CandidateDetailsResponseModel
    Id : int
    Name : string?
    DateOfBirth : DateTime
    Address : string?
    Town : string?
    Country : string?
    PostCode : string?
    PhoneHome : string?
    PhoneMobile : string?
    PhoneWork : string?
    CreatedDate : DateTime
    UpdatedDate : DateTime
```

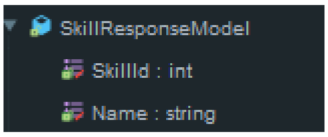

```
CandidateListItemResponseModel
    Id : int
    Name : string?
    DateOfBirth : DateTime
    Town : string?
    Phone : string?
```

Firefish.Core.Models.Skill.Requests


CandidateSkillRequestModel
CandidateId : int
SkillId : int

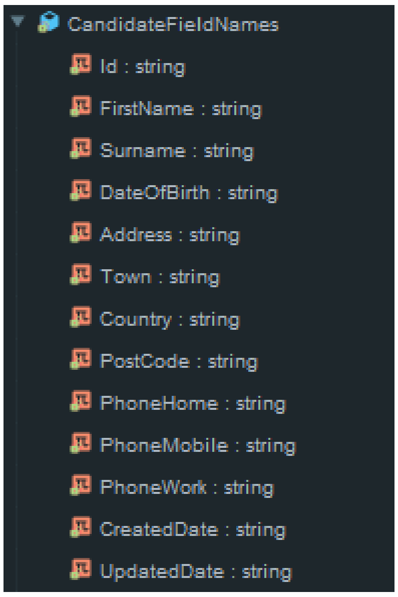
Firefish.Core.Models.Skill.Responses


CandidateSkillResponseModel
CandidateSkillId : int
SkillId : int
Name : string

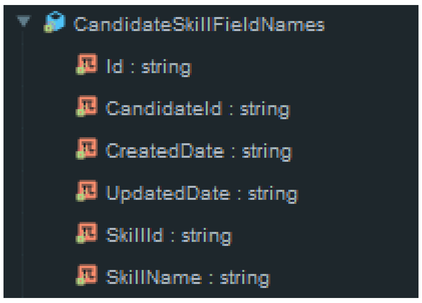

SkillResponseModel
SkillId : int
Name : string

Firefish.Infrastructure

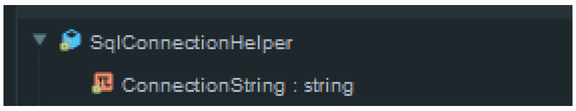
Firefish.Infrastructure.Helpers



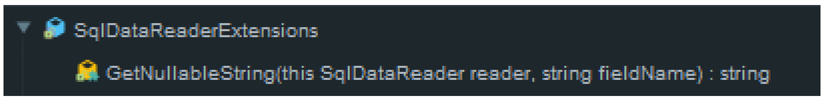
```
CandidateFieldNames
    Id : string
    FirstName : string
    Surname : string
    DateOfBirth : string
    Address : string
    Town : string
    Country : string
    PostCode : string
    PhoneHome : string
    PhoneMobile : string
    PhoneWork : string
    CreatedDate : string
    UpdatedDate : string
```



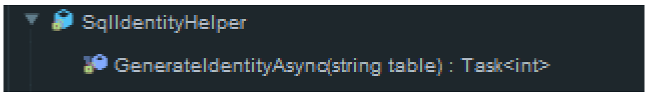
```
CandidateSkillFieldNames
    Id : string
    CandidateId : string
    CreatedDate : string
    UpdatedDate : string
    SkillId : string
    SkillName : string
```



```
SqlConnectionHelper
    ConnectionString : string
```

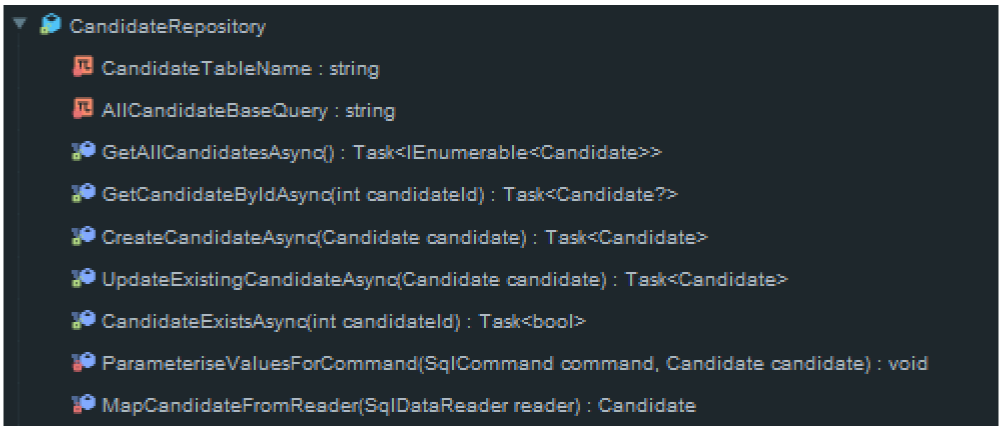


```
SqlDataReaderExtensions
    GetNullableString(this SqlDataReader reader, string fieldName) : string
```

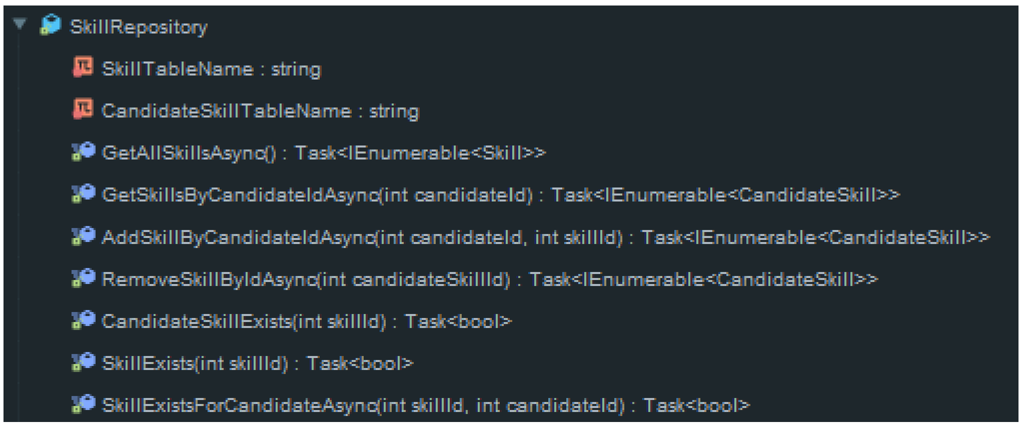


```
SqlIdentityHelper
    GenerateIdentityAsync(string table) : Task<int>
```

Firefish.Infrastructure.Repositories

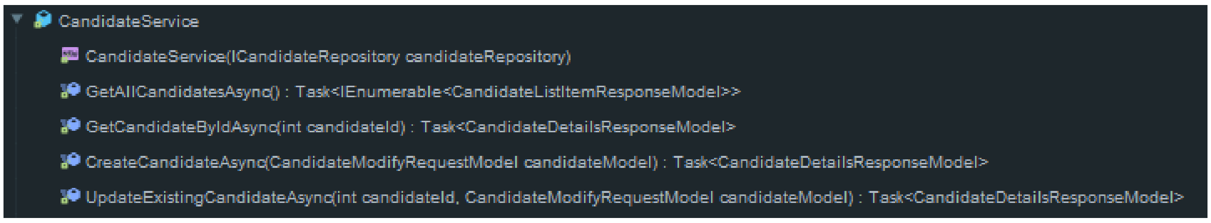


```
CandidateRepository
    CandidateTableName : string
    AllCandidateBaseQuery : string
    GetAllCandidatesAsync() : Task<IEnumerable<Candidate>>
    GetCandidateByIdAsync(int candidateId) : Task<Candidate?>
    CreateCandidateAsync(Candidate candidate) : Task<Candidate>
    UpdateExistingCandidateAsync(Candidate candidate) : Task<Candidate>
    CandidateExistsAsync(int candidateId) : Task<bool>
    ParameteriseValuesForCommand(SqlCommand command, Candidate candidate) : void
    MapCandidateFromReader(SqlDataReader reader) : Candidate
```

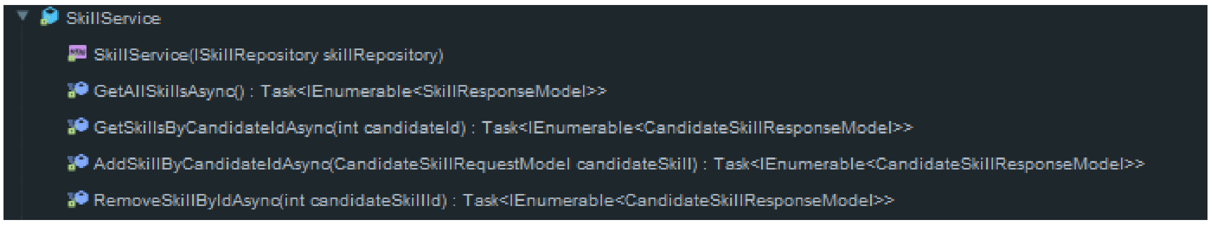


```
SkillRepository
    SkillTableName : string
    CandidateSkillTableName : string
    GetAllSkillsAsync() : Task<IEnumerable<Skill>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkill>>
    AddSkillByCandidateIdAsync(int candidateId, int skillId) : Task<IEnumerable<CandidateSkill>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkill>>
    CandidateSkillExists(int skillId) : Task<bool>
    SkillExists(int skillId) : Task<bool>
    SkillExistsForCandidateAsync(int skillId, int candidateId) : Task<bool>
```

Firefish.Infrastructure.Services



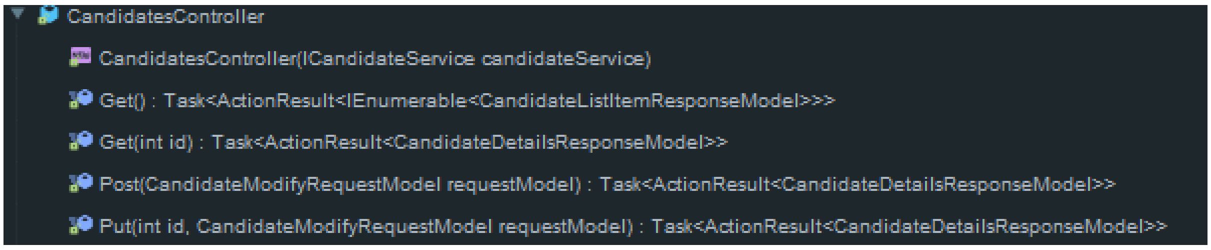
```
CandidateService
    CandidateService(ICandidateRepository candidateRepository)
    GetAllCandidatesAsync() : Task<IEnumerable<CandidateListItemResponseModel>>
    GetCandidateByIdAsync(int candidateId) : Task<CandidateDetailsResponseModel>
    CreateCandidateAsync(CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
    UpdateExistingCandidateAsync(int candidateId, CandidateModifyRequestModel candidateModel) : Task<CandidateDetailsResponseModel>
```



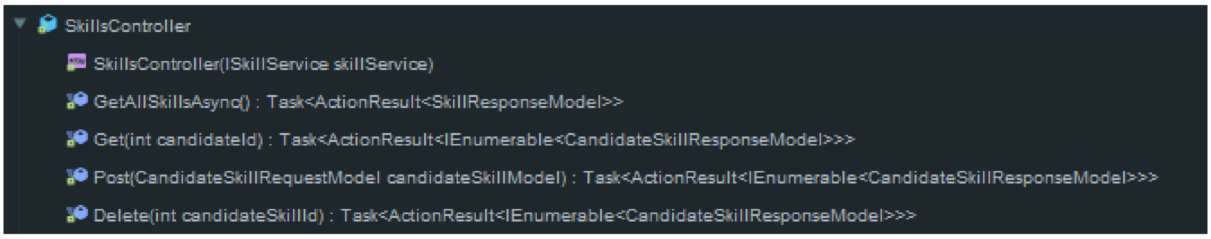
```
SkillService
    SkillService(ISkillRepository skillRepository)
    GetAllSkillsAsync() : Task<IEnumerable<SkillResponseModel>>
    GetSkillsByCandidateIdAsync(int candidateId) : Task<IEnumerable<CandidateSkillResponseModel>>
    AddSkillByCandidateIdAsync(CandidateSkillRequestModel candidateSkill) : Task<IEnumerable<CandidateSkillResponseModel>>
    RemoveSkillByIdAsync(int candidateSkillId) : Task<IEnumerable<CandidateSkillResponseModel>>
```

Firefish.API

Firefish.API.Controllers



```
CandidatesController
    CandidatesController(ICandidateService candidateService)
    Get() : Task<ActionResult<IEnumerable<CandidateListItemResponseModel>>>
    Get(int id) : Task<ActionResult<CandidateDetailsResponseModel>>
    Post(CandidateModifyRequestModel requestModel) : Task<ActionResult<CandidateDetailsResponseModel>>
    Put(int id, CandidateModifyRequestModel requestModel) : Task<ActionResult<CandidateDetailsResponseModel>>
```



```
SkillsController
    SkillsController(ISkillService skillService)
    GetAllSkillsAsync() : Task<ActionResult<SkillResponseModel>>
    Get(int candidateId) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
    Post(CandidateSkillRequestModel candidateSkillModel) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
    Delete(int candidateSkillId) : Task<ActionResult<IEnumerable<CandidateSkillResponseModel>>>
```