Архитектурен проект

„UrbanWebSystem“

Автори: *Денис Табутов, Любослав Балев,*

Симона Нешкова, Симона Кирилова

Съдържание:

I. Въведение

"UrbanWebSystem" е уеб-базирана платформа, създадена с цел подобряване на комуникацията между гражданите и местната администрация. Системата предоставя лесен достъп до информация за проекти, предложения и възможност за активно участие чрез подаване на предложения, обсъждания и гласуване.

Документът представя архитектурата на системата, като дефинира основните ѝ компоненти, изгледи и функционалности.

Участници:

- **Фронтенд разработчик:** Отговаря за изграждането на потребителския интерфейс, интеграцията на API за взаимодействие с бекенда и оптимизацията на производителността.
- **Бекенд разработчик:** Разработва API за управление на данни и функционалности на приложението, включително обработка на предложения и статуса на проектите.
- **Разработчик на база данни:** Проектира структурата на базата данни и осигурява ефективност, сигурност и надеждност.
- **Софтуерен архитект:** Определя структурата на основните компоненти и техните връзки, избира подходящи технологии, осигурява сигурността и оптимизацията на системата.
- **QA специалист:** Провежда тестове за осигуряване на качеството, включително функционално тестване и ръчно тестване на потребителския интерфейс.

II. Предназначение на архитектурния проект

1. **Обхват**

Архитектурният проект обхваща всички аспекти на разработката, като включва:

- **Планиране:** Оценка на изискванията и нуждите на системата, определяне на ресурси и времеви рамки за всяка фаза на разработката.
- **Анализ на изискванията:** Определяне на основните функционалности на системата, включително подаване на предложения, обсъждания и управление на проекти.
- **Дизайн:** Избор на архитектура, бази данни и технологии, проектиране на потребителския интерфейс и инфраструктурни нужди.
- **Програмиране:** Разработка на клиентски и сървърни компоненти, включително база данни и API.
- **Тестване:** Функционално тестване на приложението.
- **Внедряване:** Избор на платформа за разгръщане на системата и мониторинг след пускане.

2. Актьори

- **Граждани:** Основни потребители, които подават предложения и участват в платформата.
- **Администратори:** Използват системата за управление на съдържание и анализ на предложенията.
- **Технически екип:** Отговорен за разработка и поддръжка.

3. Използвани термини и символи

- ***Razor Webpages:*** Това са динамични уеб страници, използвани в ASP.NET приложения, които комбинират HTML и C# код за генериране на съдържание на сървъра и визуализация на клиента.
- ***Bootstrap****:* CSS и JavaScript библиотека за създаване на отзивчиви (responsive) уеб интерфейси.
- ***ASP.NET***: Фреймуърк на Microsoft за разработка на уеб приложения.
- ***MSSQL Server****:* Релационна база данни на Microsoft, използвана за съхранение и управление на данни.
- ***Entity Framework Core (EFCore)****:* инструмент за работа с базата данни чрез обектно-ориентирано програмиране.

4. **Източници**

- ISO/IEC 42010: Системна и софтуерна архитектура.
- Техническа документация за .NET Framework.

III. Архитектурен обзор

1. Use-case изглед

Цел и обхват:
Use-case изгледът дефинира основните взаимодействия между различните видове потребители и системата. Той дава цялостна представа за функционалностите, които системата ще поддържа от гледна точка на потребителите и техните цели.

Основни Use-case сценарии:

- **Граждани:**
  - ***Подаване на предложения:*** Гражданите могат да въвеждат заглавие, описание и категория за нови инициативи.
  - ***Обсъждане:*** Потребителите коментират и дават обратна връзка по вече съществуващи предложения.
  - ***Гласуване***: Потребителите гласуват за предложения, за да приоритизират най-важните инициативи.
  - ***Проследяване на проекти***: След одобрение гражданите могат да следят статуса и напредъка на проектите.

- **Администратори:**

- ***Управление на предложения***: Преглеждат, одобряват или отхвърлят предложения, като добавят коментари.
- ***Управление на проекти:*** Одобряват предложения за реализация и ги превръщат в изпълними проекти.
- ***Генериране на отчети:*** Изготвят отчети за популярността на предложенията, резултатите от гласуванията и напредъка на проектите.

2. Логически изглед

Цел и обхват:
Логическият изглед описва структурните елементи на системата като модули, класове и компоненти и техните зависимости. Този изглед ще даде на разработчиците информация за разпределението на функционалностите и основните слоеве на системата.

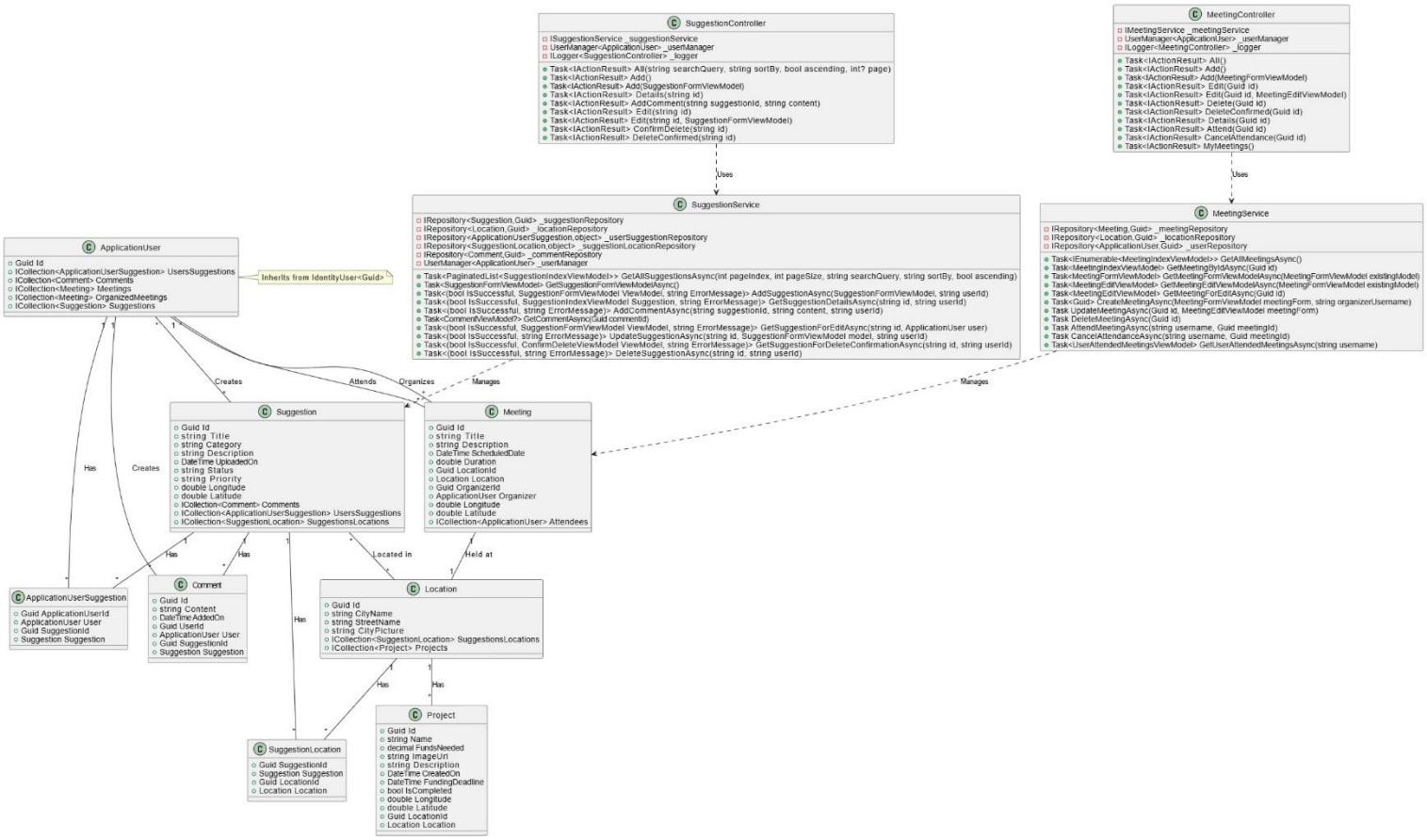
Основни компоненти:

- **Модели (Models)** - представляват основните класове, дефиниращи структурата на данните и релациите в базата данни**:**
    - ***ApplicationUser:*** Поддържа връзки с обекти като предложение, среща и коментар.
    - ***Meeting:*** Проследява подробностите за срещата, участниците и организаторите.
    - ***Comment:*** Представлява потребителски коментари, свързани с предложения или срещи.
    - ***Suggestion:*** Улавя предоставените от потребителите идеи и предложения за градско развитие.
    - ***ApplicationUserSuggestion:*** Управлява връзките много към много между потребители и предложения.

- **Services -** осигурява комуникацията между контролерите и базата данни чрез предоставяне на методи за обработка на данни и бизнес правила**:**

- ***SuggestionService:*** Управлява логиката за предложенията (създаване, валидиране, модериране).
- ***MySuggestionServise:*** Управлява логиката за предложенията на логнатия потребител
- ***ProjectService*****:** Обработва данните за проектите (статус, напредък).
- ***UserService:*** Управлява данни за потребителите и техните роли.
- ***MeetingService:*** Управлява срещите.
- ***LocationService:*** Отговаря за логиката на локациите

- **Контролери (Controllers) -** Всеки контролер съответства на конкретен домейн (напр. местоположение, среща или проект) и управлява взаимодействието между потребителския интерфейс и основните услуги**:**
  - ***MeetingController:*** Управление на срещите (създаване, редактиране, изтриване и преглед).
  - ***SuggestionController:*** Управление на потребителските предложения.
  - ***MySuggestionController:*** Управляване на предложенията направени от логнатия потребител (преглед, създаване, редактиране, изтриване)
  - ***ProjectController:*** Управлява информацията за проектите.
  - ***HomeController:*** За управление на home екран
  - ***LocationConroller:*** За управление на локациите



3. Процесен изглед

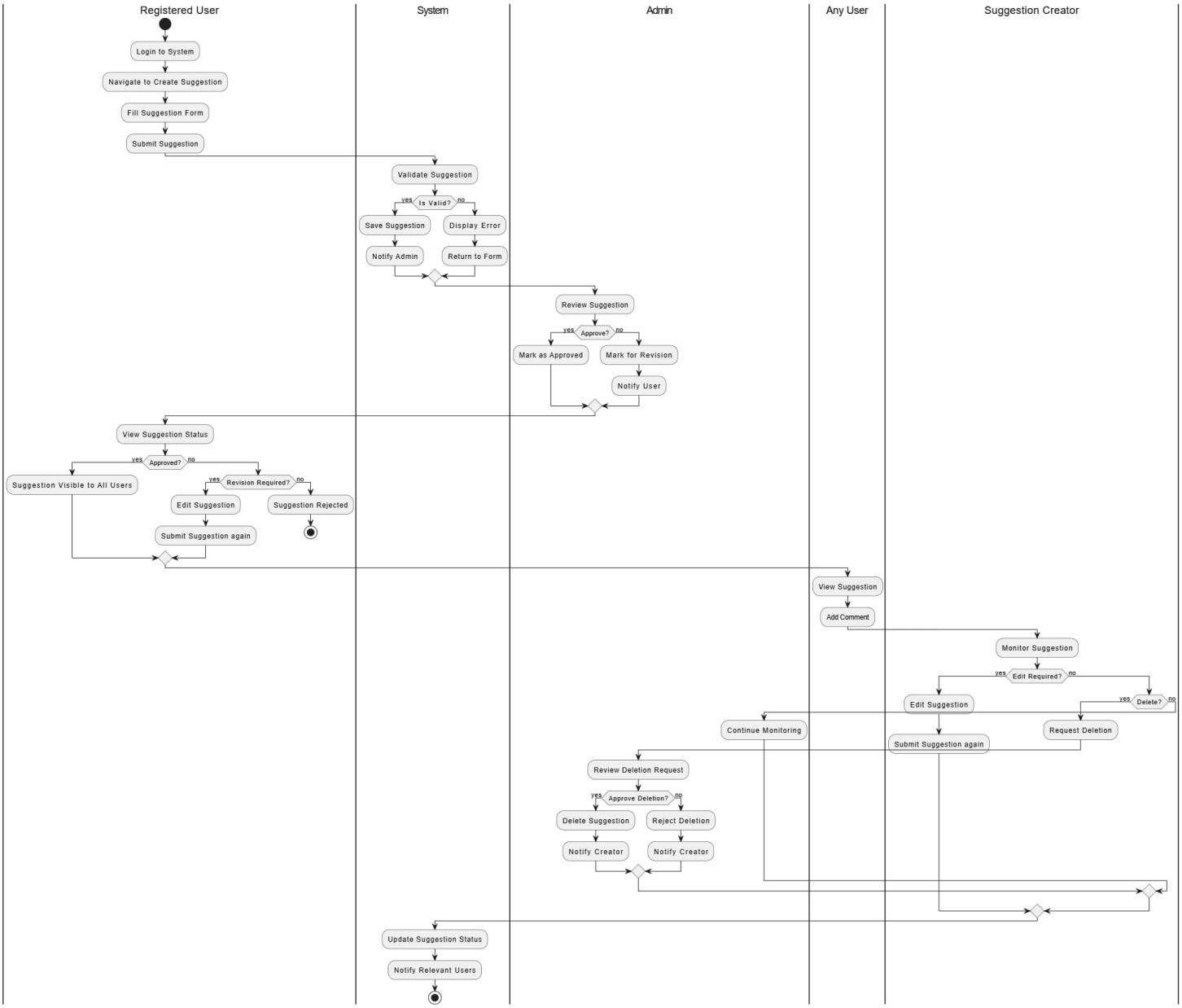
Цел и обхват:
Процесният изглед показва потока на информация и последователността на действията в системата, както и взаимодействието между компонентите в реализацията на ключови функционалности.

Основни процеси:

- **Подаване на предложение:**
  - Гражданинът попълва формуляр за ново предложение.
  - Данните се валидират и съхраняват в базата.
  - Администраторът преглежда предложението и го одобрява или отхвърля.

- **Гласуване:**
  - Потребителите разглеждат списъка с предложения и избират тези, които подкрепят.
  - Системата регистрира гласовете и обновява резултатите в реално време.

- **Управление на проект:**
  - Одобреното предложение се трансформира в проект.
  - Администраторът задава срокове, бюджети и задачи.
  - Потребителите проследяват напредъка чрез визуализации и отчети.



4. Изглед на данните

Цел и обхват:
Изгледът на данните се фокусира върху структурата на базата от данни и логическите връзки между нейните таблици. Той ще осигури ясна представа за това как ще бъдат съхранявани и свързани данните в системата.

Основни таблици:

- ***Users:*** Съхранява информация за потребителите, включително ID, име, имейл и роля.
- ***Suggestions:*** Представя подадените предложения с ID, заглавие, описание и статус.
- ***Comments:*** Съхранява коментари с ID, текст и референция към потребителя.
- ***Meetings:*** Съхранява подробностите за срещата, ID, участниците и организаторите.
- ***Projects:*** Описва одобрените проекти с ID, статус и прогрес.

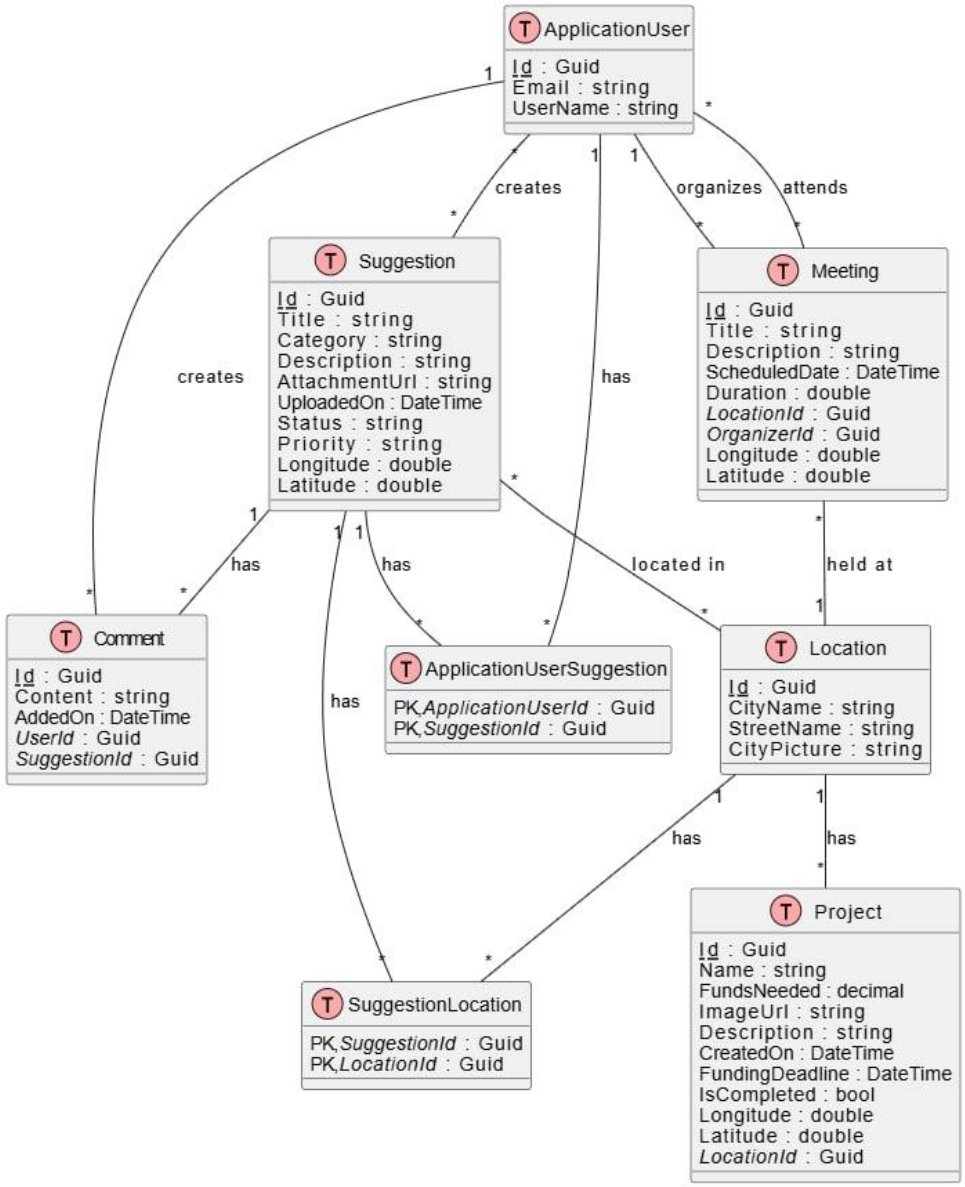
Релации:

Релации 1-М (Едно към много):

- **AspNetUsers -> Suggestions**: Един потребител (**AspNetUsers**) може да създаде много предложения (**Suggestions**).
- **Suggestions -> Comments**: Едно предложение (**Suggestions**) може да има много коментари (**Comments**).
- **Locations -> SuggestionsLocations**: Едно местоположение (**Locations**) може да е свързано с много предложения чрез междинната таблица **SuggestionsLocations**.

Релации М-М (Много към много):

- **AspNetUsers <-> Suggestions**: Представена чрез междинната таблица **UsersSuggestions**, която свързва много потребители с много предложения.
- **AspNetUsers <-> Meetings**: Реализирана чрез **ApplicationUserMeeting**, която позволява много потребители да участват в много срещи.



5. Изглед на внедряването

Цел и обхват:
Изгледът на внедряването представя разпределението на софтуера върху хардуерни ресурси и мрежовата архитектура.

Инфраструктурни компоненти:

- ***Клиентска част:*** Реализирана чрез Razor Webpages и Bootstrap
- ***Сървърна част:*** ASP.NET

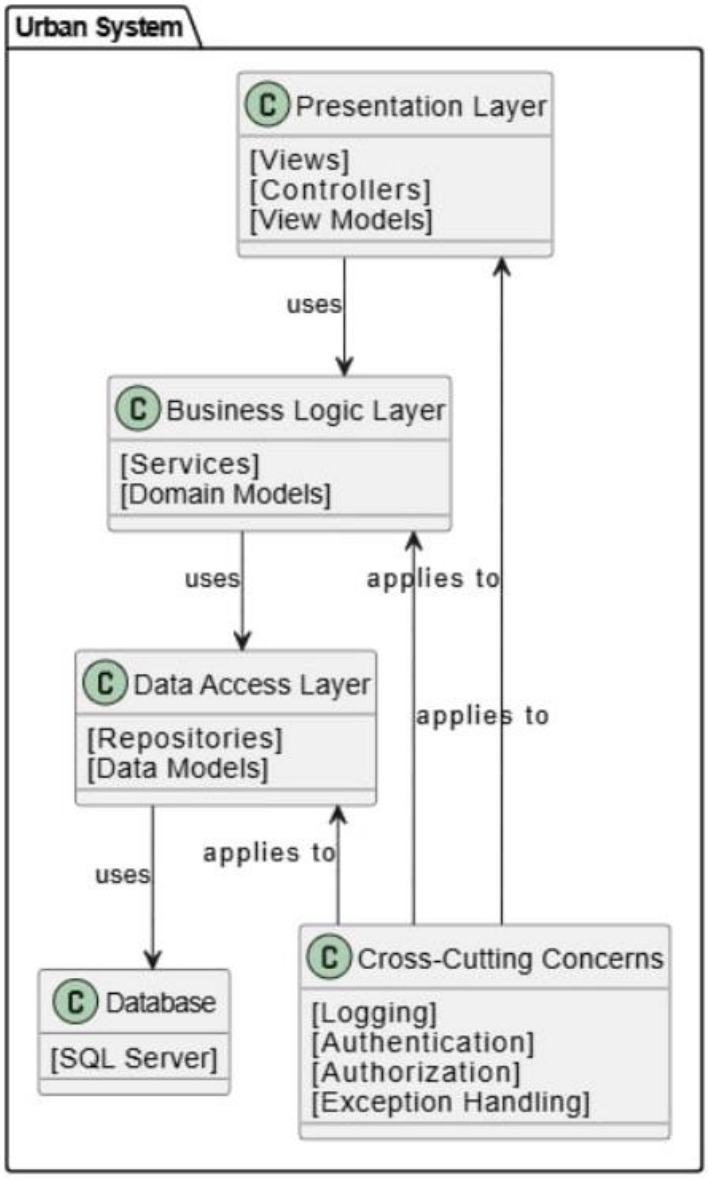
- ***База данни:*** MSSQL Server, EFCore за проектиране и комуникация с базата данни

6. Изглед на имплементацията

Цел и обхват: Изгледът на имплементацията описва детайли относно реализацията на системата, включително използваните технологии и стандарти за кодиране. Този изглед ще даде представа на разработчиците за подхода към имплементация и организационната структура на кода.

Основни слоеве на архитектурата:

- **Презентационен слой (Presentation Layer):** Отговаря за взаимодействието с потребителя, включително потребителския интерфейс (UI), уеб формулярите и API за комуникация, които предават заявки към бизнес логиката.
- **Слой на бизнес логиката (Business Logic Layer):** Съдържа сървисите и бизнес правилата, които реализират логиката за управление на предложенията, проектите и потребителските взаимодействия.
- **Слой за достъп до данни (Data Access Layer):** Включва SQL заявки, които осигуряват интерфейс за комуникация с базата данни.
- **Слой за база от данни (Database Layer):** Съдържа самата база данни, в която се съхранява информация за потребителите, предложенията и проектите.



IV. Нефункционални изисквания

1. Достъпност

- Системата трябва да бъде достъпна за гражданите и администрацията чрез стандартни уеб браузъри.
- Поддръжка на 24/7 достъпност с резервно копиране на данните на всеки 24 часа.
- Основните функции да са налични дори при частични откази.

2. Разширяемост

- Архитектурата позволява добавяне на нови модули или функционалности без преработка на основния код.
- Ясна слоеста структура, която улеснява внедряването на нови контролери и услуги.

3. Производителност

- Оптимално време за отговор под 1 секунда при нормална натовареност.
- Кеширане на често използвани данни, за да се минимизират заявките към базата данни.

4. Сигурност

- Шифроване на пароли и лични данни в базата.
- HTTPS за защита на комуникацията между клиента и сървъра.

5. Възможност за тестване

- Модулни тестове за всяка основна функционалност на системата.
- Логическите компоненти да бъдат разделени, за да се улесни индивидуалното тестване.

6. Използваемост

- Потребителският интерфейс трябва да бъде интуитивен, с ясно обозначени функции и лесна навигация.
- Поддръжка на мобилни устройства с адаптивен дизайн.
- Минимизиране на времето за обучение на потребителите чрез предоставяне на насоки в интерфейса.