|  |  |
| --- | --- |
| |  | | --- | | **УНИВЕРЗИТЕТ У НОВОМ САДУ**  **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**  **НОВИ САД**  **Департман за рачунарство и аутоматику**  **Одсек за рачунарску технику и рачунарске комуникације**  **Дипломски рад**  **Кандидат: Тамара Антић**  **Број индекса: RA 215/2019**  **Тема рада:**  **Ментор рада: др Владимир Димитриески, ванредни професор**  **Нови Сад, месец, година** | |

|  |  |  |  |
| --- | --- | --- | --- |
| Редни број, **РБР**: | |  | |
| Идентификациони број, **ИБР**: | |  | |
| Тип документације, **ТД**: | | Монографска документација | |
| Тип записа, **ТЗ**: | | Текстуални штампани материјал | |
| Врста рада, **ВР**: | | Дипломски рад | |
| Аутор, **АУ**: | | Тамара Антић | |
| Ментор, **МН**: | | др Владимир Димитриески, ванредни професор | |
| Наслов рада, **НР**: | |  | |
| Језик публикације, **ЈП**: | | Српски / ћирилица | |
| Језик извода, **ЈИ**: | | Српски | |
| Земља публиковања, **ЗП**: | | Република Србија | |
| Уже географско подручје, **УГП**: | | Војводина | |
| Година, **ГО**: | | 2024 | |
| Издавач, **ИЗ**: | | Ауторски репринт | |
| Место и адреса, **МА**: | | Нови Сад; трг Доситеја Обрадовића 6 | |
| Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога) | |  | |
| Научна област, **НО**: | | Електротехничко и рачунарско инжењерство | |
| Научна дисциплина, **НД**: | | Примењене рачунарске науке и информатика | |
| Предметна одредница/Кqучне речи, **ПО**: | | Базе података и информациони системи | |
| **УДК** | |  | |
| Чува се, **ЧУ**: | | У библиотеци Факултета техничких наука, Нови Сад | |
| Важна напомена, **ВН**: | |  | |
| Извод, **ИЗ**: | |  | |
| Датум прихватања теме, **ДП**: | |  | |
| Датум одбране, **ДО**: | |  | |
| Чланови комисије, **КО**: | Председник: |  |
|  | Члан: |  | Потпис ментора |
|  | Члан, ментор: | др Владимир Димитриески, ванр. проф. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Accession number, **ANO**: | |  | |
| Identification number, **INO**: | |  | |
| Document type, **DT**: | | Monographic publication | |
| Type of record, **TR**: | | Textual printed material | |
| Contents code, **CC**: | | Bachelor Thesis | |
| Author, **AU**: | | **Tamara Antić** | |
| Mentor, **MN**: | | Vladimir Dimitrieski, PhD, Associate Professor | |
| Title, **TI**: | |  | |
| Language of text, **LT**: | | Serbian | |
| Language of abstract, **LA**: | | Serbian | |
| Country of publication, **CP**: | | Republic of Serbia | |
| Locality of publication, **LP**: | | Vojvodina | |
| Publication year, **PY**: | | 2024 | |
| Publisher, **PB**: | | Author’s reprint | |
| Publication place, **PP**: | | Novi Sad, Dositeja Obradovica sq. 6 | |
| Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes) | |  | |
| Scientific field, **SF**: | | Electrical Engineering | |
| Scientific discipline, **SD**: | | Computer Engineering, Engineering of Computer Based Systems | |
| Subject/Key words, **S**/**KW**: | | Databases and information systems | |
| **UC** | |  | |
| Holding data, **HD**: | | The Library of Faculty of Technical Sciences, Novi Sad, Serbia | |
| Note, **N**: | |  | |
| Abstract, **AB**: | |  | |
| Accepted by the Scientific Board on, **ASB**: | |  | |
| Defended on, **DE**: | |  | |
| Defended Board, **DB**: | President: |  |
|  | Member: |  | Menthor's sign |
|  | Member, Mentor: |  |  |

**Садржај**

[1. Увод 4](#_Toc157367304)

[2. Опис реалног система 6](#_Toc157367305)

[2.1 Класе корисника и њихове карактеристике 6](#_Toc157367306)

[2.2 Функционални захтјеви 7](#_Toc157367307)

[2.2.1 Пријава на систем 7](#_Toc157367308)

[2.2.2 Симулација уласка/изласка радника из пословног објекта 7](#_Toc157367309)

[2.2.3 Надгледање параметара на нивоу пословног објекта 7](#_Toc157367310)

[2.2.4 Надгледање параметара на нивоу канцеларије 7](#_Toc157367311)

[2.2.5 Систем за провјеру безбједности 7](#_Toc157367312)

[2.2.6 Слање упозорења свим запосленим 8](#_Toc157367313)

[2.2.7 Приказ, креирање, брисање и модификовање података 8](#_Toc157367314)

[3. Опис модела података 9](#_Toc157367315)

[3.1 Релациона база података 9](#_Toc157367316)

[3.1.1 Модел логичке шеме базе података 10](#_Toc157367317)

[3.1.2 Модел имплементационе шеме базе података 11](#_Toc157367318)

[3.1.3 Опис табела у шеми базе података 11](#_Toc157367319)

[3.2 База података временских серија 15](#_Toc157367320)

[3.3 Анализа одабира базе у зависности од контекста података 18](#_Toc157367321)

[4. Архитектура система 20](#_Toc157367322)

[5. Пословна логика 23](#_Toc157367323)

[5.1 Имплементација аларм система 23](#_Toc157367324)

[5.1.1 Провјере (енг. checks) у InfluxDB бази података 25](#_Toc157367325)

[5.1.2 Дио пословне логике аларм система на серверској страни 27](#_Toc157367326)

[6. Опис апликативног рјешења 30](#_Toc157367327)

[6.1 Апликација за симулирање уласка/изласка из пословне зграде 30](#_Toc157367328)

[6.2 Апликација за надгледање канцеларијског простора 32](#_Toc157367329)

[6.2.1 Пријава у систем 32](#_Toc157367330)

[6.2.2 Надгледање параметара на нивоу зграде 32](#_Toc157367331)

[6.2.3 Надгледање параметара на нивоу канцеларије 33](#_Toc157367332)

[6.2.4 Приказ, креирање, брисање и модификовање канцеларија, радника, типова сензора, сензора и инсталација 34](#_Toc157367333)

[7. Приједлози за даља усавршавања 40](#_Toc157367334)

[7.1 Безбједност система 40](#_Toc157367335)

[7.2 Евиденција записа (енг. *log*) 41](#_Toc157367336)

[7.3 Оптимизација базе података 41](#_Toc157367337)

[8. Закључак 42](#_Toc157367338)

[9. Литература 44](#_Toc157367339)

[Списак кориштених скраћеница 46](#_Toc157367340)

[Биографија 47](#_Toc157367341)

# Увод

Тема проучавања овог дипломског рада фокусирана је на базама података, са конкретним акцентом на базе података временских серија (енг. *Time Series Database)*.

У књизи „*Time Series Databases New Ways to Store and Access Data“* аутора *Ted Dunning* и *Ellen Friedman* [1] прва мантра је представљена реченицом *„Collect your data as if your life depends on it!”.* Ова реченица јасно указује на вриједност података и важност исправног чувања истих.

У овом раду, база података временских серија је искориштена за систематично и ефикасно сакупљање, чување и надгледање података добијених путем сензора распоређених у канцеларијама пословне зграде. Три кључна параметра - температура, влажност и ниво угљен моноксида - пажљиво се прате у реалном времену, пружајући особљу одржавања детаљан увид у околности унутар објекта. Свака канцеларија је опремљена са по три сензора, чинећи систем свеобухватним и прецизним у детекцији потенцијалних проблема.

Једна од кључних функционалности овог система је визуализација података путем графикона, што омогућава радницима да лако прате параметре на нивоу зграде и сваке појединачне канцеларије. Ова транспарентност и брза доступност информација пружају особљу одржавања могућност да предузму одговарајуће кораке у реалном времену, и тиме драстично смањује вријеме реакције одговорних лица на опасну ситуацију.

Нарочито значајно, систем не само да чува податке, већ и активно реагује на неочекиване вриједности мјерења. Уколико измјерене вриједности изађу из опсега дефинисаних нормалних вриједности, систем аутоматски обавјештава особље одржавања. У случају екстремних вриједности, када постоји пријетња по сигурност, радници су непосредно обавијештени о потреби евакуације, чиме се обезбјеђује брза и ефикасна реакција на потенцијалне опасности.

У наредним поглављима овог рада, размотрићемо концепт рјешења, теоријске основе технологија кориштених у имплементацији као и детаље имплементације система.

# Опис реалног система

У овом поглављу се истражује реални систем, описује начин његовог функционисања кроз идентификована правила пословања, као и захтјеви који су постављени пред систем током фазе спецификације корисничких захтјева.

## Класе корисника и њихове карактеристике

Систем разликује три групе корисника:

1. Администратор,
2. Радник одржавања система и
3. Радник.

Корисници типа администратор су овлаштена лица која имају право приступа свим функционалностима система. Администратор је задужен да стање у реалном систему евидентира у информациони систем, односно да у систему региструје сваку канцеларију из пословне зграде, забиљежи сваки инсталиран сензор и слично. Све у циљу да информациони систем вјеродостојно осликава реални, и омогући олакшавање надгледања очитаних вриједности са сензора као и аутоматизацију система за слање обавјештења.

Корисници типа радник одржавања система су људи који су задужени за надгледање очитавања са сензора и којима је омогућено слање наређења о евакуацији свим радницима у згради. Због вјеродостојности информација и стабилности система, радник одржавања система нема право приступа административном дијелу софтвера, тачније функционалности 2.2.7 из поглавља Функционални захтјеви.

Корисници типа радник представљају запослене у пословном објекту који се надгледа. Они имају најограниченија права приступа, која се своде на посмарање вриједности очитаних параметара.

## Функционални захтјеви

### Пријава на систем

Корисници који су претходно регистровани у систем од стране администратора могу да се пријаве на систем користећи своју мејл адресуи лозинку.

Корисник типа радник може да приступи функционалностима 2.2.2, 2.2.3, 2.2.4 и 2.2.5.

Корисник типа радник одржавања система може да приступи функционалностима 2.2.2, 2.2.3, 2.2.4, 2.2.5, 2.2.6.

Корисник типа администратор може да приступ свим функционалностима система.

### Симулација уласка/изласка радника из пословног објекта

Систем треба да омогући кориснику избор пословног простора и симулацију уласка или изласка запосленог из претходно одабраног пословног простора. Корисник би требао да може да одабере конкретног запосленог, а систем би затим требао да генерише сценарио уласка или изласка. Ова функционалност такође би требало да обезбиједи логичку провјеру, као што је ограничење да корисник може да симулира излазак само за оне запослене који се у том тренутку налазе унутар пословног објекта. Додатно, систем би требао да чува податке о симулираним уласцима и изласцима, омогућавајући каснију анализу и праћење активности запослених.

### Надгледање параметара на нивоу пословног објекта

Систем треба да омогући кориснику систематичан графички приказ свих параметара (угљен моноксида, температуре и влажности ваздуха) на нивоу пословног објекта. Графикон треба да приказује вриједности параметара у реалном времену.

### Надгледање параметара на нивоу канцеларије

Систем треба да омогући кориснику систематичан графички приказ свих параметара (угљен моноксида, температуре и влажности ваздуха) на нивоу сваке канцеларије у пословном објекту. Графикони треба да приказују вриједности параметара у реалном времену.

### Систем за провјеру безбједности

Потребно је имплементирати систем за провјеру безбједности. Овај систем подразумијева константну провјеру сачуваних вриједности мјерења сензора, упоређивање са очекиваним вриједностима и обавјештавање одговорних лица уколико вриједности изађу из очекиваног опсега.

Слање обавјештења се врши на два начина:

* Обавјештење на корисничком интерфејсу

Уколико дође до било каквог одступања од опсега очекиваних вриједности на неком од сензора потребно је информисати раднике путем обавјештења на корисничком интерфејсу.

* Обавјештење путем електронске поште

Уколико се на неком од сензора забиљежи нижа вриједност од очекиване послаће се обавјештење запосленим лицима из канцеларије у којој је измјерена неочекивана вриједност путем електронске поште, али само онима од њих који се тренутно налазе у пословном објекту.

Уколико се на неком од сензора забиљежи виша вриједност од очекиване послаће се упозорење, путем електронске поште свим запосленим лицима који се тренутно налазе у пословном објекту.

Алгоритам за слање упозорења је детаљно објашњен у поглављу 5.

### Слање упозорења свим запосленим

Поред аутоматског слања обавјештења које врши систем по правилима дефинисаним у претходном функционалном захтјеву, потребно је омогућити раднику из одржавања који надгледа све параметре у згради да кликом на дугме пошаље свим запосленима наређење о евакуацији уколико примијети необична очитавања.

У том случају свим запосленима унутар зграде се шаље електронска пошта којом се налаже радницима да хитно напусте зграду.

### Приказ, креирање, брисање и модификовање података

Потребно је омогућити приказ, брисање, модификовање и креирање нових података о канцеларијама, сензорима, типовима сензора, инсталацији сензора и радницима.

# Опис модела података

Ово поглавље детаљно разлаже модел података на којем се темељи информациони систем. У имплементацији софтвера кориштене су двије различите базе података. Релациона база података је кориштена за чување података о свим ентитетима релевантним за систем, као што су зграде, канцеларије, радници, сензори итд. Са друге стране, нерелациона база података, тачније база података временских серија је кориштена за чување свих мjерења са сензора, као и догађаја уласка и изласка радника из зграде. Овакав приступ омогућава оптимално управљање различитим врстама података у складу са њиховим специфичностима и потребама система.

## Релациона база података

У оквиру система релациона база података је кориштена за чување и руковање са информација о наведеним ентитетима у систему:

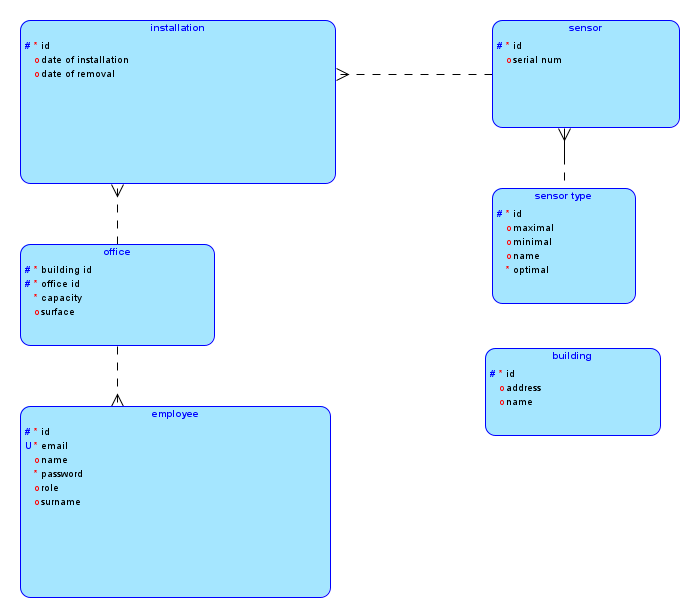
* зграда,
* радник,
* тип радника,
* инсталација,
* канцеларија,
* улога,
* сензор,
* име сензора и
* тип сензора.

За поједностављено дефинисање шеме базе података, примијењен је *Java Persistence API (JPA)* унутар *Spring Boot* оквира. Овај приступ ефикасно генерише потребну шему базе података на основу модела кориштеног у *Java* апликацији. Важно је нагласити да изостанак потребе за ручним креирањем детаљне шеме потиче из њене нискe комплексности. Шема базе података овог система лако и идентично може бити креирана путем *ЈPA*, обезбјеђујући при том конзистентност и усклађеност са захтјевима апликације. Овакав приступ значајно смањује вријеме израде шеме.

У наредном поглављу представљена је логичка и имплементациона шема базе података. Шеме су генерисане кориштењем алата *Oracle Data Modeler* од *ddl* скрипте коју је генерисао *ЈРА*. Овај корак има за циљ пружање визуалне репрезентације и детаљног објашњења шеме базе података која је основа система, доприносећи бољем разумијевању архитектуре података у оквиру система.

### Модел логичке шеме базе података

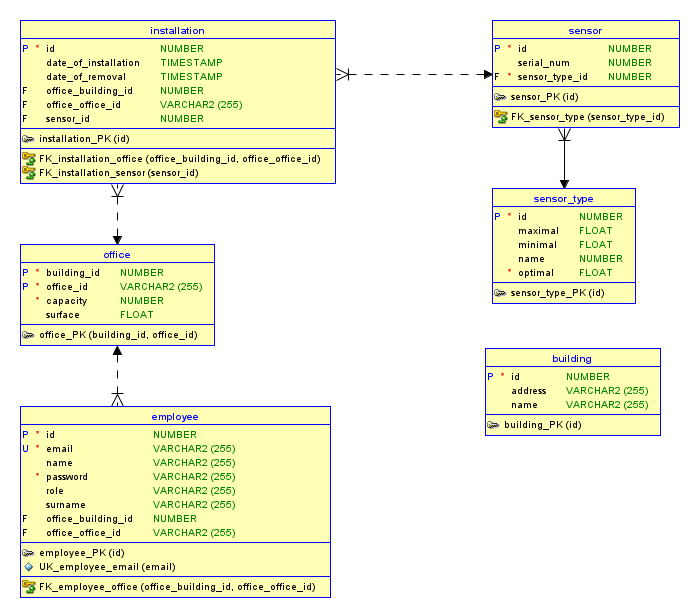
На слици 3.1.1.1 је приказана логичка шема базе података креирана помоћу *Oracle Data Modeler* алата.



*3.1.1.1 Логичка шема базе података*

### Модел имплементационе шеме базе података

На слици 3.1.2.1 је приказана имплементациона шема базе података креирана помоћу *Oracle Data Modeler* алата.



*3.1.2.1 Имплементациона шема базе података*

### Опис табела у шеми базе података

У овом поглављу представљене су спецификације свих релација, приказаних у табеларном облику са свим њиховим карактеристикама и ограничењима. Ова детаљна анализа има за циљ да пружи потпуни увид у структуру података и међусобне везе унутар система.

#### Табела *Building*

Шема релације зграда (енг. *building*) садржи основне податке о згради.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *id* | *BIGSERIAL* | Не | Јединствени идентификатор |
| *address* | *VARCHAR* (255) | Да | Адреса зграде |
| *name* | *VARCHAR* (255) | Да | Назив зграде |

|  |  |
| --- | --- |
| **Ограничење примарног кључа** | |
| **Назив ограничења** | **Колоне** |
| *PK*\_ЗГРАДА | *id* |

#### Табела *Employee*

Шема релације радник (енг. *employee*) садржи основне податке о раднику.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *id* | *BIGSERIAL* | Не | Јединствени идентификатор |
| *email* | *VARCHAR(255)* | Не | *Еmail* адреса радника |
| *name* | *VARCHAR(255)* | Не | Име |
| *surname* | *VARCHAR(255)* | Не | Презиме |
| *password* | *VARCHAR(255)* | Не | Лозинка |
| *role* | *VARCHAR(255)* | Не | Улога у систему |
| office\_id | *VARCHAR(255)* | Не | Јединствени идентификатор канцеларије у којој ради радник |

|  |  |
| --- | --- |
| **Ограничење примарног кључа** | |
| **Назив ограничења** | **Колоне** |
| *PK*\_РАДНИКА | *id* |

|  |  |  |  |
| --- | --- | --- | --- |
| **Ограничење референцијалног интегритета** | | | |
| **Назив ограничења** | **Колоне** | **Референцирана табела** | **Референциране колонe** |
| *FK*\_КАНЦЕЛАРИЈА\_ЗГРАДА | office\_building\_id | Канцеларија | *building\_id* |
| *FK*\_КАНЦЕЛАРИЈА\_КАНЦЕЛАРИЈА | office\_office\_id | Канцеларија | *office\_id* |

|  |  |
| --- | --- |
| **Ограничење јединствености** | |
| **Назив ограничења** | **Колоне** |
| *UQ*\_еmail | *email* |

#### Табела *Office*

Шема релације канцеларија (енг. *office*) садржи основне податке о канцеларији.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *building\_id* | *BIGINT* | Не | Јединствени идентификатор зграде у којој се налази канцеларија |
| *office\_id* | *VARCHAR(255)* | Не | Јединствени идентификатор канцеларије |
| *capacity* | *INTEGER* | Не | Број радних станица у канцеларији |
| *surface* | *FLOAT(53)* | Не | Површина канцеларије у квадратним метрима |

|  |  |
| --- | --- |
| **Ограничење примарног кључа** | |
| **Назив ограничења** | **Колоне** |
| *PK*\_КАНЦЕЛАРИЈА | *building\_id + office\_id* |

|  |  |  |  |
| --- | --- | --- | --- |
| **Ограничење референцијалног интегритета** | | | |
| **Назив ограничења** | **Колоне** | **Референцирана табела** | **Референциране колонe** |
| *FK*\_ЗГРАДА | *building\_id* | Зграда | *id* |

#### Табела *Sensor*

Шема релације сензор (енг. *sensor*) садржи основне податке о сензору.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *id* | *BIGSERIAL* | Не | Јединствени идентификатор |
| *serial\_num* | *BIGINT* | Не | Серијски број сензора |
| *sensor\_type\_id* | *BIGINT* | Не | Јединствени идентификатор типа сензора којем припада сензор |

|  |  |
| --- | --- |
| **Ограничење примарног кључа** | |
| **Назив ограничења** | **Колоне** |
| *PK*\_СЕНЗОР | *id* |

|  |  |  |  |
| --- | --- | --- | --- |
| **Ограничење референцијалног интегритета** | | | |
| **Назив ограничења** | **Колоне** | **Референцирана табела** | **Референциране колонe** |
| *FK*\_СЕНЗОР\_ТИП | *sensor\_type\_id* | Тип сензора | *id* |

#### Табела *Sensor\_Type*

Шема релације тип сензора (енг. *sensor type*) садржи основне податке о типу сензора.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *id* | *BIGSERIAL* | Не | Јединствени идентификатор |
| *maximal* | *FLOAT(53)* | Не | Максимална сигурна вриједност |
| *minimal* | *FLOAT(53)* | Не | Минимална сигурна вриједност |
| *name* | *SMALLINT* | Не | Назив типа сензора |
| *optimal* | *FLOAT(53)* | Не | Оптимална вриједност |

|  |  |
| --- | --- |
| **Ограничење примарног кључа** | |
| **Назив ограничења** | **Колоне** |
| *PK*\_ТИП\_СЕНЗОРА | *id* |

#### Табела *Installation*

Шема релације инсталација (енг. *Installation)* садржи основне податке о инсталацији одређеног сензора.

|  |  |  |  |
| --- | --- | --- | --- |
| **Колоне у табели** | | | |
| **Назив колоне** | **Тип података** | ***Nullable*** | **Опис колоне** |
| *id* | *BIGSERIAL* | Не | Јединствени идентификатор |
| *date\_of\_installation* | *TIMESTAMP(6)* | Не | Датум и вријеме инсталације сензора |
| *date\_of\_removal* | *TIMESTAMP(6)* | Да | Датум и вријеме уклањања сензора |
| *office\_building\_id* | *BIGINT* | Не | Јединствени идентификатор зграде у којој је инсталиран сензор |
| *office\_office\_id* | *VARCHAR(255)* | Не | Јединствени идентификатор канцеларије у којој је инсталиран сензор |
| *sensor\_id* | *BIGINT* | Не | Јединствени идентификатор сензора |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ограничење примарног кључа** | | | | |
| **Назив ограничења** | | **Колоне** | | |
| *PK*\_ИНСТАЛАЦИЈА | | *id* | | |
| **Ограничење референцијалног интегритета** | | | | | |
| **Назив ограничења** | **Колоне** | | **Референцирана табела** | **Референциране колонe** | |
| *FK*\_КАНЦЕЛАРИЈА\_ЗГРАДА | office\_building\_id | | Канцеларија | *building\_id* | |
| *FK*\_КАНЦЕЛАРИЈА\_КАНЦЕЛАРИЈА | office\_office\_id | | Канцеларија | *office\_id* | |
| *FK*\_СЕНЗОР | *sensor\_id* | | Сензор | *id* | |

## База података временских серија

Временске серије представљају мjерења или догађаје који се систематски прате, надгледају и агрегирају током одређеног временског периода. Ови подаци могу да обухватају различите врсте аналитичких информација, укључујући метрике сервера, перформансе апликација, мрежне податке, сензорске податке, догађаје, кликове, трансакције на тржишту и многе друге. Кључна карактеристика која издваја временске серије од обичних података је стална потреба за постављањем питања о њима током времена. Идентификација скупа података као временских серија једноставно се спроводи провjером да ли једна од оса представља временски континуум.

*InfluxDB* [2] пружа корисницима могућност прикупљања, складиштења, упита, обраде и визуализације сирових података високе прецизности, као и агрегираних и узоркованих података. Платформа *InfluxDB* организује временске серије у структуираном формату, где се на највишем нивоу налази назив мјерења. Затим слиједи скуп парова кључ/вриједност, познатих као тагови (енг. *tags*), који описују метаподатке, и кључеви/вриједности стварних вриједности, познатих као поља (енг. *fields*). Вриједности поља у *InfluxDB* могу бити у облику *boolean, int64, float64* или *string*. На крају, сваком скупу вриједности придружен је временски печат. Над подацима се врши упит по мјерењу, таговима, пољима и временском опсегу.[1]

Основни концепти у *InfluxDB* бази података су: [2]

* **Поља** представљају обавезну компоненту структуре података. Без присуства поља, подаци не могу бити смјештени у *InfluxDB*. Битно је напоменути да поља нису индексирана, што значи да упити који користе вриједности поља као филтере морају прегледати све вриједности које се подударају са осталим условима у упиту, што може довести до смањења перформанси. Уопштено, поља не би требала садржавати често претраживане метаподатке.
* **Тагови** су опциони, али препоручљив дио структуре података у I*nfluxDB*-у. За разлику од поља, тагови су индексирани, што резултује бржим упитима над таговима, што их чини идеалним за чување често претраживаних метаподатака. Тагови се састоје од кључева и вриједности који се чувају као *string*-ови.
* **Мјерење** дјелује као контејнер за тагове, поља и временску колону. Име мјерења је опис података који се чувају у повезаним пољима. Концептуално, мјерење је слично табели у *SQL* бази података. Једно мјерење може припадати различитим политикама задржавања *(*енг. *retention policies)*.
* **Контејнер** (енг. *Bucket*) је именовано мјесто гдје се складиште подаци временских серија. Сваки контејнер има политику задржавања, тј. временски период током кога се чува свака тачка података.
* **Политика задржавања података** *(*енг. *Retention Policies)* описује колико дуго *InfluxDB* чува податке и колико копија тих података је смјештено у бази.
* **Тачка** представља појединачни запис који има четири компоненте: мјерење, скуп тагова, скуп поља и временски печат. Свака тачка је јединствено идентификована својом серијом и временским печатом.
* **База података** чува све ове компоненте. *InfluxDB* база података служи као логички контејнер за кориснике, политике задржавања, континуалне упите и временске серије податка. *InfluxDB* не захтијева стикту шему базе података, што значи да је једноставно додавати нова мјерења, тагове и поља у било које вријеме, олакшавајући рад са временским серијама података.

У овом информационом систему су употријебљена два контејнера. Први контејнер под називом *airSensor-RT*, служи за чување података о мјерењима са сензора. Подаци се у овај контејнер смјештају периодичним извршавањем задатка (енг. *task*) названим „*Collect air sensor data - real Time Simulation*“. Извршавањем кода скрипте, наведене испод, подаци се повлаче са званичног *Air Sensor Data Set*-a и чувају у контејнер *airSensor-RT.*

import "influxdata/influxdb/sample"

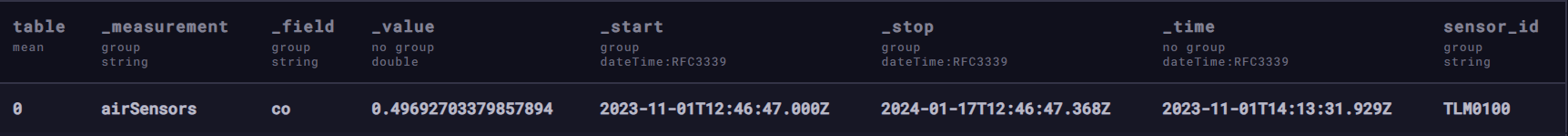
import "date"

sample.data(set: "airSensor")

|> map(fn: (r) => ({r with \_time: date.add(d: 15m, to: r.\_time)}))

|> to(bucket: "airSensor-RT")

У пољу овог контејнера се налазе информације о очитаним вриједностима са сензора, док су у тагу смјештени бројеви канцеларија у којима су сензори постављени. На слици 3.2.1 се налази примјер једног податка, тачке у *airSensor-RT* контејнеру*.*

 *Слика 3.2.1 Примјер једне тачке у airSensor-RT контејнеру*

Други контејнер, назван *buildingEvents*, служи за складиштење догађаја уласка/изласка радника из пословне зграде. Сваки пут када се симулира улазак или излазак радника из зграде, у овај контеинер се сачува нова тачка са потребним инфомрацијама. У пољу овог контејнера се налази информација о правцу кретања радника, док су у тагу смјештени идентификациони бројеви радника, као што је илустровано на слици 3.2.2. Ова структура података је дизајнирана имајући у виду честе упите система, који траже информације о посљедњој активности сваког радника. Оптимизација упита је постигнута употребом тага за чување идентификације радника. Примjер тога је сљедећи упит:

from(bucket: \"" + influxDBConnection.getBucket() + "\")

|> range(start: -24h)

|> sort(columns: [\"\_time\"], desc: false)

|> group(columns: [\"employeeId\"])

|> last()

|> filter(fn: (r) => r.\_value == \"in\")

 *Слика 3.2.2 Примјер једне тачке у buildingEvents контејнеру*

## Анализа одабира базе у зависности од контекста података

Одабир оптималне базе података за развој апликације представља важну одлуку која значајно утиче на перформансе, скалабилност и одржавање система. Потребно је систематски размотрити више кључних аспеката како би се донијела информисана одлука у складу са специфичностима пројекта.

У почетку, кључно је сагледати структуру података која ће се чувати у систему. Релационе базе података пружају чврст и јасно дефинисан модел путем табела, док нерелационе базе омогућавају већу флексибилност у приступу подацима. Одлука о моделу података требала би одражавати специфичне потребе апликације и комплексност информација које ће се чувати.

Када је ријеч о перформансама и скалабилности, треба размотрити очекиване захтјеве и будући раст апликације. Нерелационе базе података, посебно оне које подржавају хоризонтално скалирање, често пружају боље перформансе у условима повећаног оптерећења у односу на традиционалне релационе базе, које се често ослањају на вертикално скалирање.

Питање трансакција и конзистентности података такође је битно. Ако је неопходна потпуна конзистентност, релационе базе које подржавају *ACID* трансакције могу бити оптималан избор. Међутим, у ситуацијама гдје је брзина извршавања приоритет, нерелационе базе података које толеришу одређени компромис у конзистентности могу бити бољи избор.

Различите врсте упита које ће апликација често извршавати такође треба узети у обзир приликом одлучивања између релационих и нерелационих база података. Релационе базе показују добре перформансе у сложеним упитима, док нерелационе базе омогућавају флексибилност у раду са обимним подацима кроз једноставније упите.

Кориштење релационе базе података за ентитете наведене у поглављу 3.1, као што су зграде, канцеларије и радници, у развијаном информационом систему образложава се њиховом природном структуром и захтјевима за обраду података. Релациона база података идеално се усклађује са прецизно структурираним подацима ових ентитета који учествују у основним операцијама читања, брисања, креирања и модификовања. Овакав избор обезбијеђује конзистентност, трајност и изолованост података, што је неопходно за стабилан и ефикасан информациони систем.

Такође, за овакве ентитете се не очекује уношење екстремних количине података и стога се предвиђа да ће систем управљања базом података успјешно обрађивати ове податке, обезбијеђујући при том добре перформансе система.

Са друге стране, за чување података о мјерењима сензора и догађајима уласка/изласка радника из пословне зграде изабрана је база података са временским серијама.

Кључно питање које ће јасно указати на потребу за кориштењем базе података са временским серијама јесте: Да ли је у интересу система чување историјске промјене вриједности или искључиво тренутних вриједности?

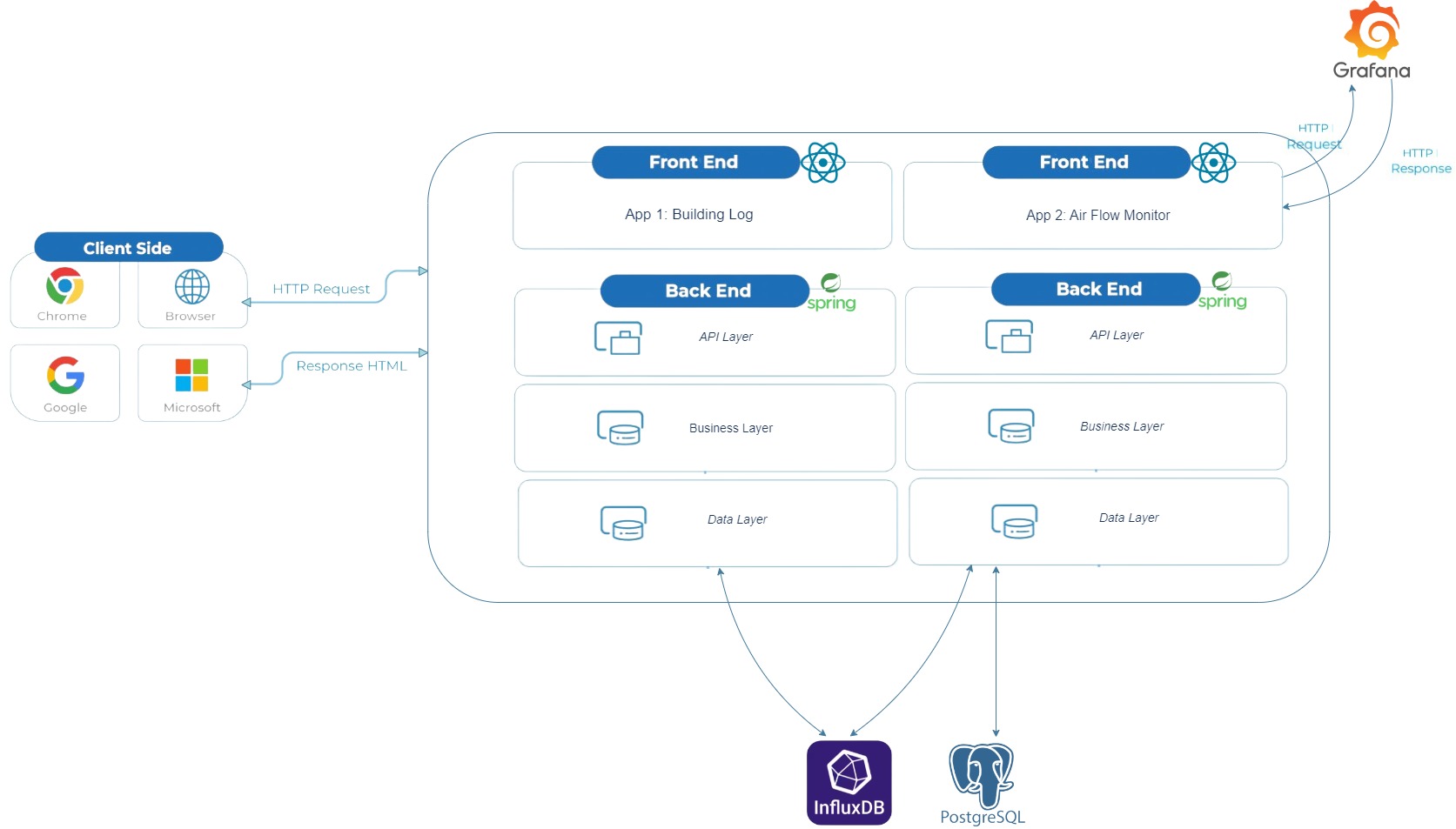
За податке о мјерењима сензора јасно је да је од интереса да се трајно сачува промјена очитаних вриједности у времену. У овом контексту се не ради измјена постојећих вриједности у бази, него се очитавања у времену чувају, акумулирају и касније обрађују. То нас доводи до сљедећег битног аспекта у овој бази података а то је анализа података.

Базе података временских серија представљају изузетно ефикасан инструмент за анализу података, посебно у контексту великог обима акумулираних података. Због њихове могућности временског обиљежавања појединачних података, ове базе обезбијеђују дубок увид у динамику компоненти током времена. Анализом међусобног утицаја компоненти и откривањем шаблона током продужених временских периода, стварају се корисне информације и закључци.

Избор базе података временских серија у контексту чувања догађаја уласка/изласка радника из пословне зграде произилази из низа могућности детаљне анализе података. Ова одлука обезбијеђује темељ за складиштење свих релевантних података током времена, стварајући могућност за будућу надоградњу система кроз имплементацију специфичних упита који ће анализирати податке о уласцима и изласцима радника. Тренутно, ови упити нису имплементирани, али постоји потенцијал за њихово касније увођење.

# Архитектура система

Квалитетно осмишљена архитектура система омогућава успостављање хармоничне интеграције различитих компоненти, обезбијеђујући стабилност, скалабилност и оптималну оперативну ефикасност. Ово поглавље ће пажљиво анализирати кључне карактеристике архитектуре система имплементираног у оквиру ове студије.



*Слика 4.1 Дијаграм архитектуре система*

Имплементиран систем се састоји од двије одвојене апликације, *BuildingLog* и *AirFlowMonitor* веб апликације. У наставку ће бити објашњене компоненте из којих се састоје обе од ових апликација.

На дијаграму архитектуре система (слика 4.1) приказане су кључне компоненте које чине *BuildingLog* апликацију. Ова апликација је специфично дизајнирана за симулацију улазака и излазака људи из пословне зграде. Клијентски дио апликације пружа корисницима једноставан приступ свим функционалностима система. Када корисник интерагује са интерфејсом, на примјер, кликом на одређено дугме, клијентски дио апликације шаље *REST* захтјев према серверском дијелу апликације.

Серверски дио апликације *BuildingLog* је у комуникацији са *InfluxDB* базом података. Приликом обраде захтјева извршавају се одговарајући упити над *InfluxDB* базом података како би се добиле неопходне информације.

Све наведене компоненте система су прецизно повезане, чиме се постиже координисано функционисање *BuildingLog* апликације.

У оквиру имплементираног система, *AirFlowMonitor* апликација игра кључну улогу, обезбијеђујући аутентификованим корисницима могућност надгледања очитаних вриједности са сензора ваздуха у пословној згради, као и обавјештавање радника о неочекиваним очитавањима. Слично *BuildingLog* апликацији, *AirFlowMonitor* такође посједује засебан клијентски и серверски дио.

Клијентски дио апликације *AirFlowMonitor* визуализује податаке о очитаним вриједностима са сензора, приказује актуелна обавјештења и омогућава кориснику приступ функционалностима система у зависности од улоге корисника у систему. Комуникација између клијентског и серверског дијела апликације *AirFlowMonitor* одвија се путем *REST* захтјева. Клијентски дио апликације шаље одређене параметре серверском дијелу, који затим извршава одговарајуће упите над *InfluxDB* или *PostgreSQL* базом података, како би добио релевантне податке.

У контексту визуализације података, клијентски дио *AirFlowMonitor*-а остварује везу са *Grafana* [3] алатом, који омогућава генерисање графикона и визуализацију података. Клијентски дио шаље упит *Grafana* алату, просљеђујући јој параметре, а *Grafana* на основу просљеђених вриједности генерише одговарајући графикон који се затим приказује на корисничком интерфејсу.

Важно је напоменути да комуникација између компоненти, осим неком акцијом корисника, може да буде иницирана и појавом одређених вриједности у бази података.

*InfluxDB* периодично извршава скрипту која провјерава да ли су све вриједности очитавања очекиване. Уколико се појави нека неочекивана вриједност послаће се порука *REST* *endpoint*-у у контролеру серверског дијела апликације.

Затим се путем *WebSocket-а* који је успостављен између серверског и клијентског дијела просљеђује обавјештење на кориснички интерфејс. Отворен *WebSocket* канал омогућује серверском дијелу да шаље обавјештења у реалном времену на основу догађаја који се дешавају у бази података.

У процесу аутентификације и ауторизације, кориштене су библиотеке као што су *json web token* за рад са *JSON* *Web* *Token*-има, док је за слање обавјештења путем електронске поште кориштена библиотека *spring boot starter mail*.

# Пословна логика

У наставку, детаљно ће бити разматран кључни аспект имплементираног система – аларм систем. Предмет анализе биће начин функционисања овог аларм система, као и детаљи његове имплементације. Биће објашњено како систем обавештава раднике, које су специфичности његове реакције на различите догађаје и како су унапријеђене перформансе кроз имплементацију.

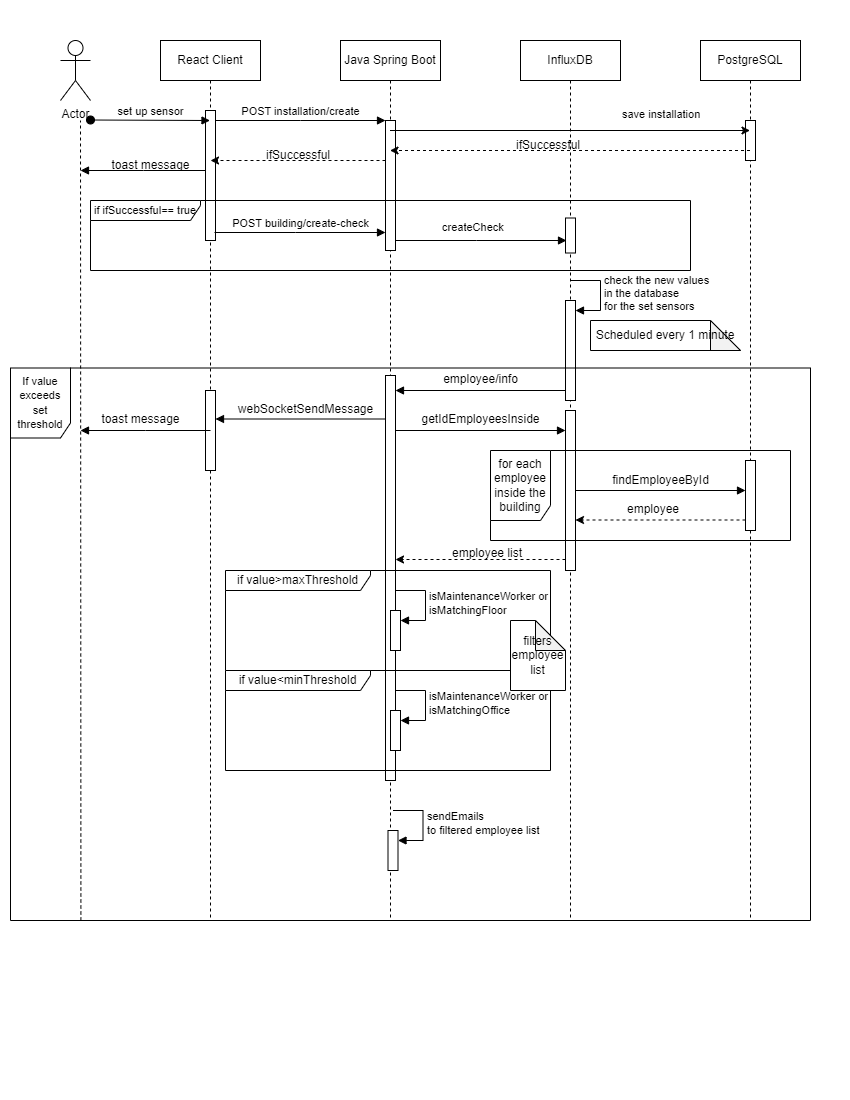
Овај аспект имплементације игра битну улогу у усмјеравању радника одржавања система ка правовременом и одговарајућем реаговању на различите податке и догађаје. Анализом детаља аларм система, ово поглавље ће пружити дубок увид у његову суштину и допринос у оптимизацији радних процеса у оквиру истраживаног информационог система.

## Имплементација аларм система

Динамички дијаграм секвенци, представљен на слици 5.1.1 састоји се из два дијела.

У првом дијелу илустрован је процес креирања новог објекта *Installation*. Објекат *Installation* садржи информације о инсталацији сензора, конкретно који сензор је постављен, у којој канцеларији и када.

Овај дио система је важан за аларм систем из разлога што се при креирању нове инсталације креира провјера (енг. *check*)у *InfluxDB* бази података, а провјере представљају основу аларм система. У наставку ће бити разложене све битне компоненте за функционисање аларм система, представљене у другом дијелу дијаграма.



*Слика 5.1.1 Дијаграм секвенци који приказује аларм систем*

На дијаграму 5.1.1 је због читљивости представљена благо поједностављена верзија алгоритма за филтрирање запослених којима ће се послати обавјештење путем електронске поште. Детаљан опис овог алгоритма за филтрирање је дат у поглављу 5.1.2.

### Провјере (енг. *Checks*) у *InfluxDB* бази података

Провјере у *InfluxDB* бази података представљају кључни механизам за надзор и евалуацију стања података у бази података временских серија. Ове провјере се користе како би се дефинисали услови или правила која се примјењују на податке, што омогућава корисницима да аутоматски детектују и реагују на нежељене или необичне догађаје у временским серијама. То укључује прекорачење прагова (енг. *thresholds*) вриједности или одсуство очекиваних података.

За исправну конфигурацију ове функционалности, неопходно је подесити три компоненте: провјере (енг. *checks*), крајње тачке за обавјештавање (енг. *notification endpoint*s) и правила за обавјештавање (енг. *notification rules*).

#### Провјере (енг. *Checks*)

Провјере представљају *Flux* скрипте које се извршавају на задатим временским интервалима. У овом систему, имплементацијом је постављен временски интервал на

1 минут, након којег се *Flux* скрипта изнова покреће. При дефинисању провјере у скрипти, потребно је задати јединствени идентификатор чека, назив и тип. Затим, треба поставити прагове (енг. *thresholds*) у односу на које систем упоређује измјерене вриједности. И наравно, потребно је дефинисати опсег података над којима ће се скрипта извршавати.

#### Прагови (енг. *Thresholds*)

Прагови представљају вриједности које се постављају као границе или услови. Ови прагови се користе за дефинисање специфичних граница или опсега вриједности унутар временских серија података. Када податак премаши или падне испод постављених прагова, провјера (енг. *check*)детектује одступања и може аутоматски покренути одговарајуће акције, као што су обавјештења или упозорења.

Такође, могуће је поставити различите услове у односу на вриједност прагова. Ови услови могу бити означени различитим статусима као што су критично (енг. *critical*)*,* упозорење (енг. *warn*)*,* информација (енг. *info*)*,* или *ok*. Оваква конфигурација дозвољава прецизно дефинисање на који начин систем треба реаговати на различите нивое одступања од дефинисаних прагова.

Систем имплементира динамичко дефинисање вриједности прагова и креирање провјера, обезбијеђујући флексибилност и адаптивност система према конкретним потребама и стандардима. У релационој бази података, за сваку инстанцу објекта тип сензора, чува се информација о најнижој и највишој дозвољеној вриједности очитавања за тај тип сензора. На примјер, истраживања о температурама у канцеларијама показују оптимални опсег између 68 и 76 фаренхајта.

Када администратор система региструје нови тип сензора, нпр. температуре, има могућност задавања најниже и највише очекиване вриједности очитавања. Ове вриједности аутоматски постају прагови у систему и користе се за креирање провјера. Администратор система такође има опцију накнадног мијењања ових вриједности, а промјене ће бити ажуриране у вриједностима прагова у *InfluxDB* бази података. Ова функционалност пружа администратору система могућност да динамички адаптира конфигурацију система у складу са промјенама и изазовима у околини.

За креирање провјера, апликација користи библиотеку *Influx Client*. Ова библиотека омогућава конекцију на *InfluxDB* базу и пружа богат скуп функционалности за ефикасно управљање и праћење података.

Креирање провјера се изводи путем класа и функција из ове библиотеке. Кориштењем класа као што су *ThresholdCheck*, *DashboardQuery*, *Threshold*, *GreaterThreshold*, *LesserThreshold*, као и функција као што су *setLevel*, *setValue*, *setQuery*, *setOrgID*, *setThresholds*, итд. апликација је у стању динамички дефинисати услове и параметре провјера.

#### Крајње тачке за обавјештавање (енг. *Notification endpoints*)

У контексту провјера у *InfluxDB* бази података, крајње тачке обавјештавања представљају цијеле одредишне локације или сервисе на које се обавјештења достављају у случају детекције одступања у подацима од стране провјера. Ови *endpoint*-и могу укључивати различите комуникационе канале као што су *Slack*, *HTTP* *endpoint*-и и *Pagerduty*. У ситуацији када је изабрана *HTTP* опција, неопходно је дефинисати *HTTP* методу, методу аутентификације и *URL*.

С обзиром на постојање аутентификације у имплементираном систему, неопходно је било ефикасно ријешити изазов методе аутентификације захтјева који пристижу од базе према дефинисаном *endpoint*-у. Ова ситуација је ријешена путем избора опције „none“ за методу аутентификације при креирању крајње тачке за обавјештења, а на серверском дијелу апликације је на бијелу листу постављена кориштена путања */employee/info*.

#### Правила за обавјештавање (енг. *Notification rules*)

Правила за обавјештавање у *InfluxDB* представљају функционалност која омогућава корисницима аутоматско покретање обавјештења или акција када се испуне одређени услови у временским серијама података. Ова функционалност представља ефикасан начин за праћење и реаговање на значајне догађаје у реалном времену.

При дефинисању правила за обавјештавање, корисници одређују провјеру (енг. *check*) као основу за обавјештење. Одабрана провјера садржи услове које подаци морају задовољити како би покренули обавјештење. Додатно, корисници конфигуришу параметре као што су „*Every*“ за одређивање учесталости провјере услова и „*For*“ за одређивање колико узастопних периода провјере услова мора бити задовољен прије него што се покрене обавјештење. Коначно, корисници одређују „*Endpoint*“ да би одредили гдје ће обавјештење бити послато. Путем ових правила, корисници могу прилагодити параметре обавјештења у складу са својим потребама, што значајно повећава аутоматизовано реаговање система на критичне или специфичне догађаје у временским серијама података.

### Дио пословне логике аларм система на серверској страни

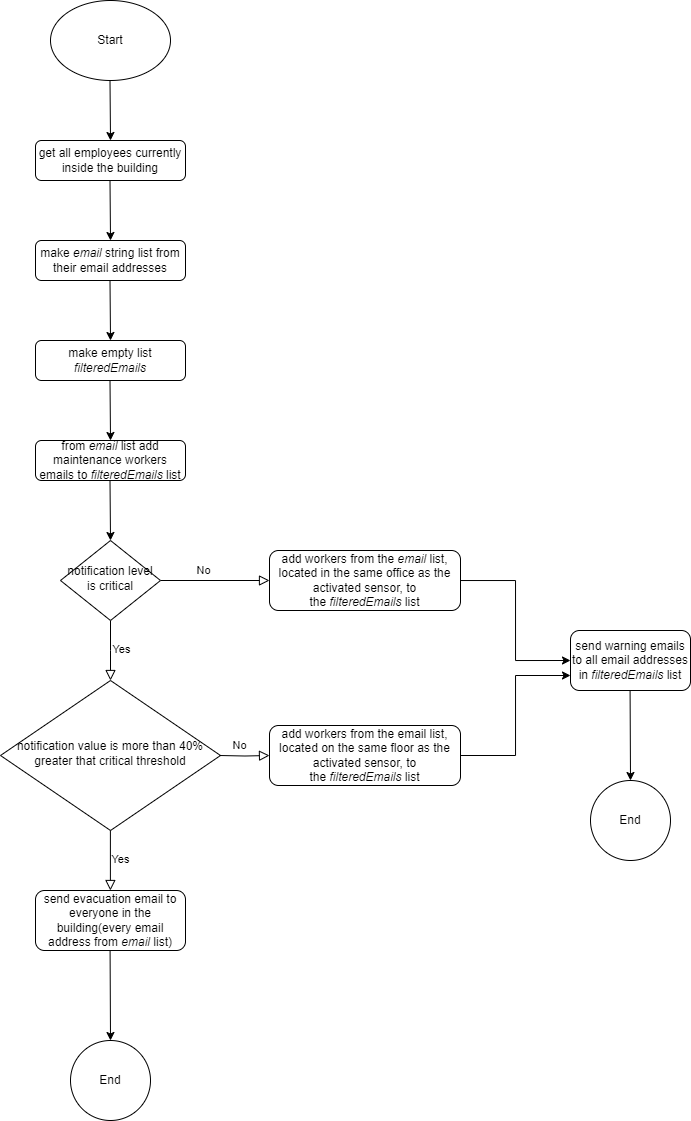
Када обавјештење буде активирано, *REST* метода на путањи */employee/inf*o прима обавјештење. Наставак описује посљедње кораке обраде обавјештења и његово дистрибуирање крајњим корисницима.

Први корак у обради обавјештења на серверској страни је парсирање пристиглог обавјештења у објекат *Notification*.

Затим се парсирано обавјештење у реланом времену просљеђује клијентској страни апликације помоћу *web socket-а*.

Узимајући у обзир да систем подржава различите улоге у систему, претпоставка је да корисник типа радник одржавања система треба да види сва обавјештења која дођу из базе података, те ће се кориснику који има ову улогу сва обавјештења приказати у виду *toast* порука. Радници осталих улога у систему биће обавјештени о само њима релевантиним обавјештењима путем електронске поште по алгоритму који ће бити објашњен у наставку.

Други корак је слање обавјештења радницима на мејл према алгоритму објашњеном дијаграмом активности на слици 5.1.2.1.



*Слика 5.1.2.1 Дијаграм активности који приказује алгоритам за слање обавјештења*

Основна идеја алгоритма јесте да одреди за кога је важно које обавјештење и у којој мјери. Па тако, уколико се деси очитавање на сензору које је мање од доње границе оптималности о томе ће се поред радника одржавања система, oбавијестити и радници у тој канцеларији у којој се десило очитавање. Овакав тип обавјештења који стигне када се појави мања вриједност од доњег прага класификују се као упозорења (енг. *warn*).

Ако се појави очитавање које је веће од горње границе оптималности одговарајућег сензора систем разликује два сценарија.

Први је ако очитане вриједности у мањој мјери прелазе горњу границу оптималности, тачније ако је прекорачење мање од 40% од горње границе оптималности. На примјер, узмимо да је опсег оптималности температуре између 68 и 76 фаренхајта, а у канцеларији на првом спрату *TLM0100* се појави очитавање 85 фаренфајта. Ова вриједност од 85 фаренхајта је за 8,44% већа од горње границе оптималности. О оваквој аномалити биће обавијештени сви радници који су тренунто у згради а раде на 1. спрату. Узима се да ова ситуација представља потенцијалну опасност и у складу са тим се упозоравају радници који раде у близини. Међутим, ако се деси прекорачење од 40% или веће, на примјер 106.4 фаренхајта, ова ситуација ће се тумачити као засигурно опасна и свим запосленим који су тренутно у згради ће се послати наређење о евакуацији.

Овај алгоритам помаже при препознавању различитих аномалија, доприноси ефикасном и детаљном управљању ситуацијама у реалном времену.

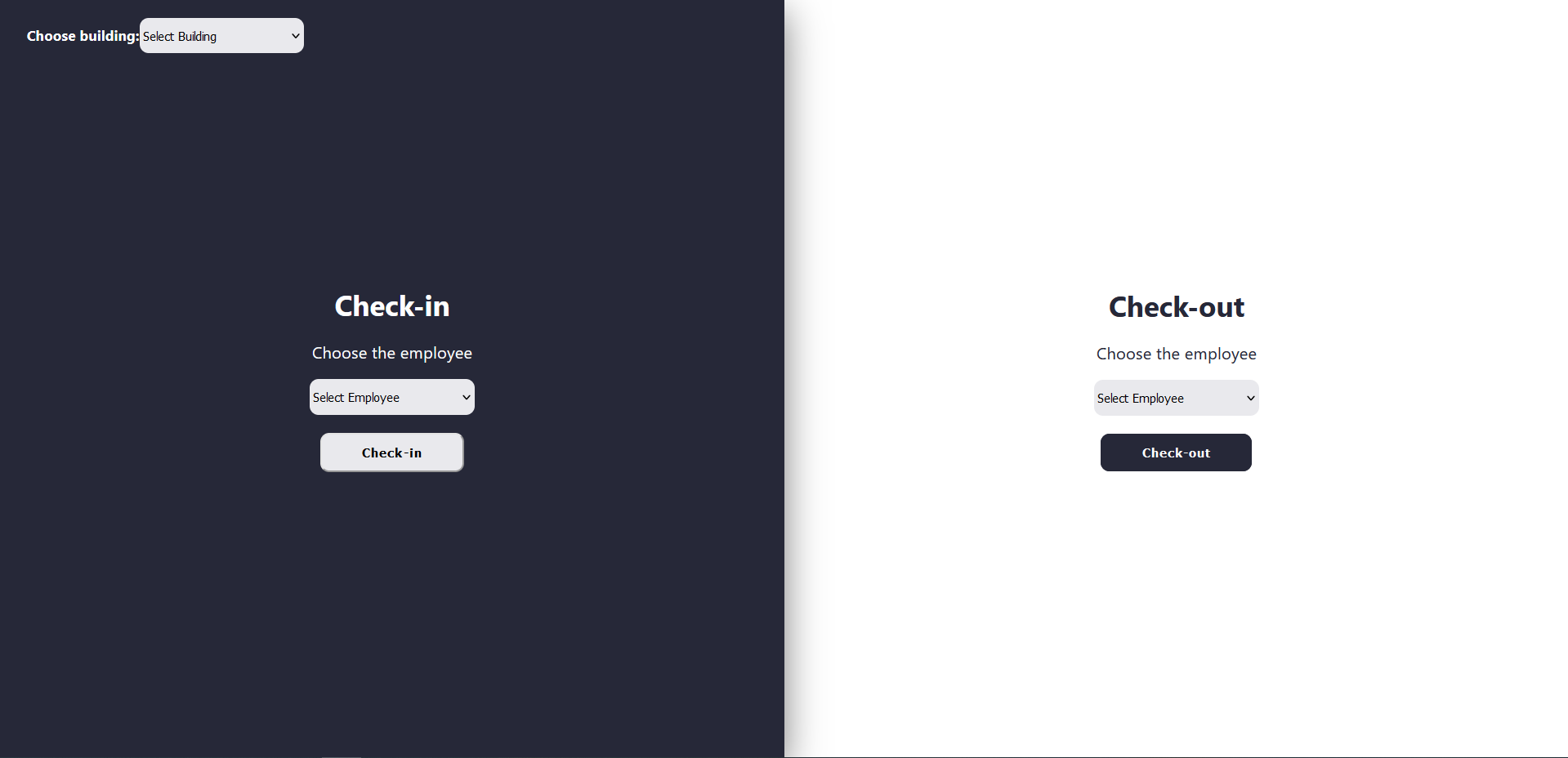
# Опис апликативног рјешења

У овом поглављу биће представљен кориснички интерфејс са освртом на појединачне цјелине и описом појединачних функционалности које су имплементиране у апликативном рјешењу.

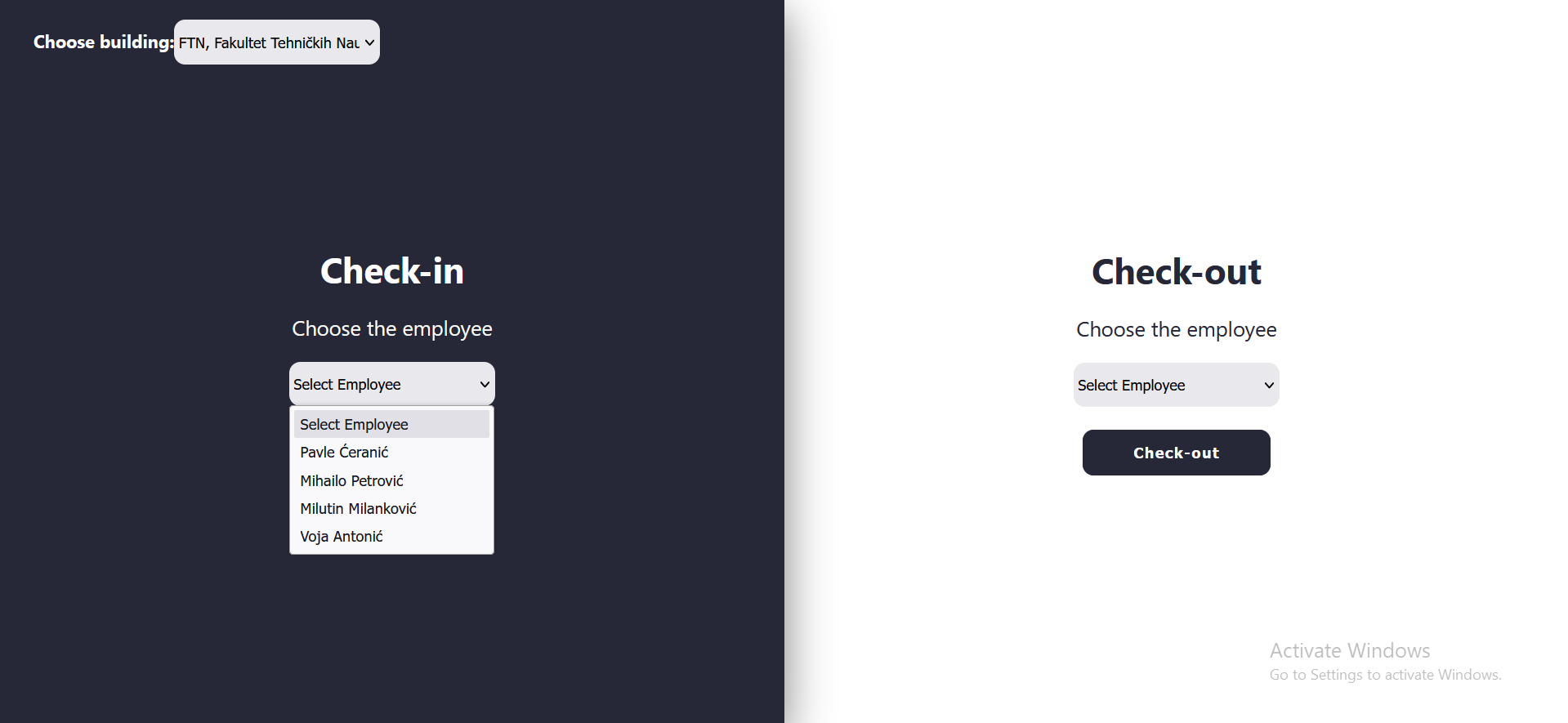
## Апликација за симулирање уласка/изласка из пословне зграде

У овом поглављу биће представљена функционалност коју подржава апликација за симулирање уласка/изласка из пословне зграде, названа *BuilidngLog* апликација.

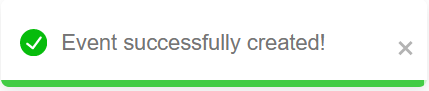
Апликација *BuildingLog* обухвата једну страницу, а дизајн ове странице приказан је на слици 6.1.1.

*Слика 6.1.1 Почетна страница BuildingLog апликације*

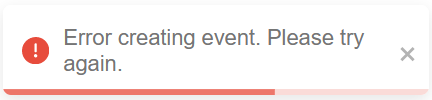
Први корак у симулацији је одабир пословне зграде из *drop down menu*-a у горњем лијевом углу. У сљедећем кораку, корисник би требао да изабере радника из *drop-down* *menu*-a, који се налази на лијевој или десној половини екрана, користећи лијеву страну ако жели симулирати улазак, а десну ако жели симулирати излазак.



*Слика 6.1.2 Приказ одабира пословне зграде и радника*

Посље избора радника акцијом која је приказана на слици 6.1.2, трећи корак укључује клик на дугме *Check In* или *Check Out*. Корисник ће бити информисан о успјешности симулације путем нотификације. У случају успјешне акције, кориснику ће бити приказана порука о успјеху илустрована на слици 6.1.3.

*Слика 6.1.3 Порука о успјешности акције*

У случају неуспјешне акције, кориснику ће бити приказана порука о грешци илустрована на слици 6.1.4.

*Слика 6.1.4 Порука о неуспјешности акције*

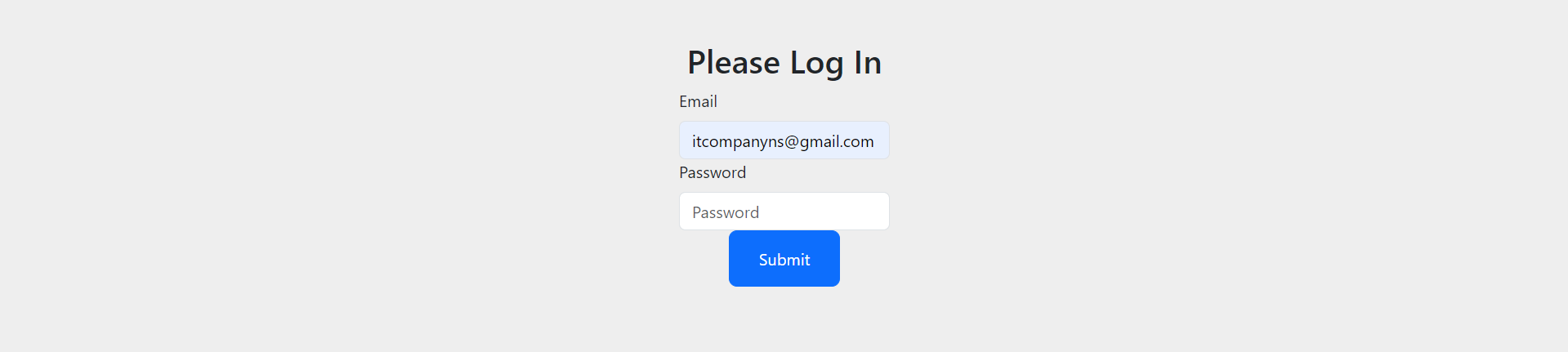
## Апликација за надгледање канцеларијског простора

У овом поглављу биће представљене функционалности које подржава апликација за надгледање канцеларијског простора, названа *AirFlowMonitor* апликација.

### Пријава у систем

Да би корисник приступио систему мора бити претходно регистрован у систем од стране администратора. Регистрован корисник приступа систему уношењем своје мејладресе и лозинке.

Форма ће кориснику сугерисати да мејладреса мора имати одређени формат и да оба поља морају бити попуњена да би акција била извршена.

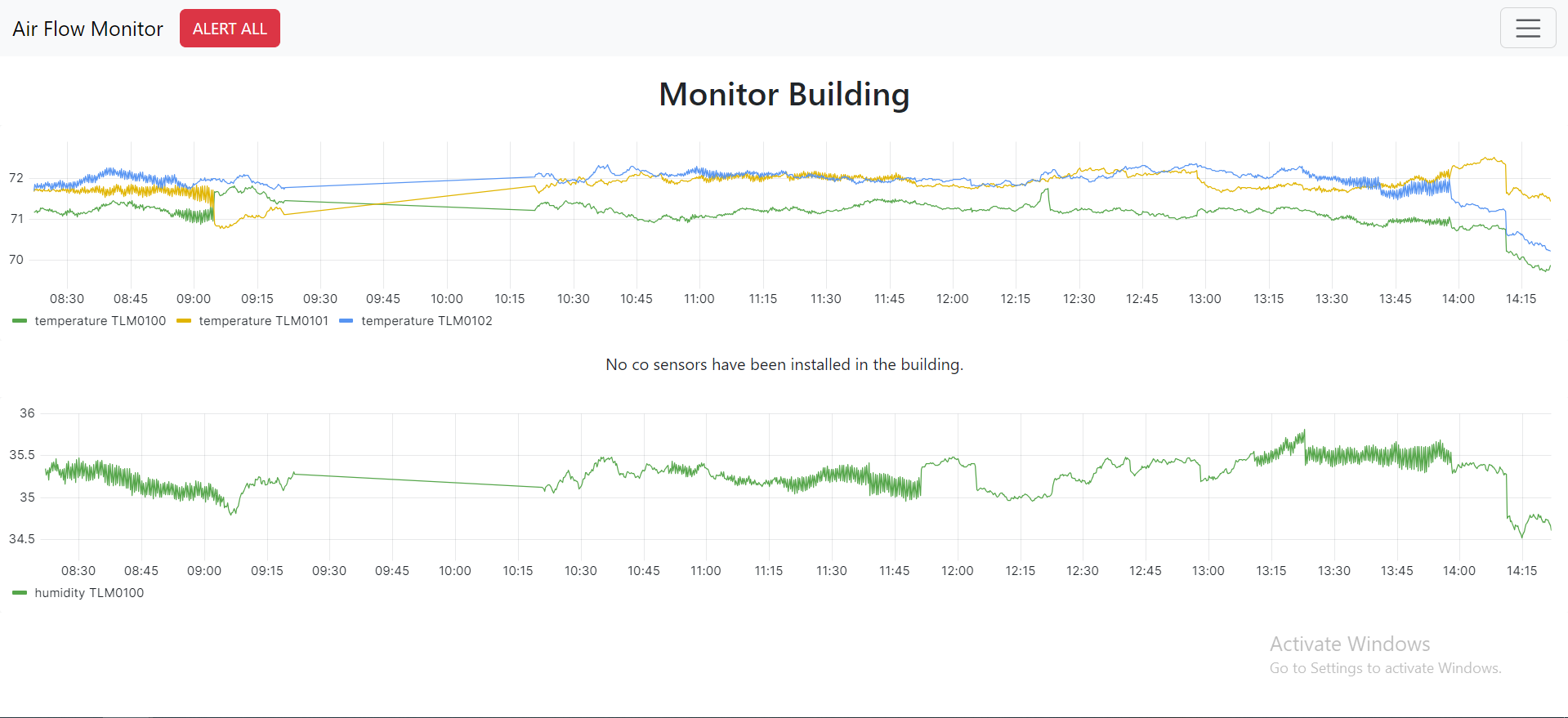


*Слика 6.2.1.1 Форма за пријаву у систем*

### Надгледање параметара на нивоу зграде

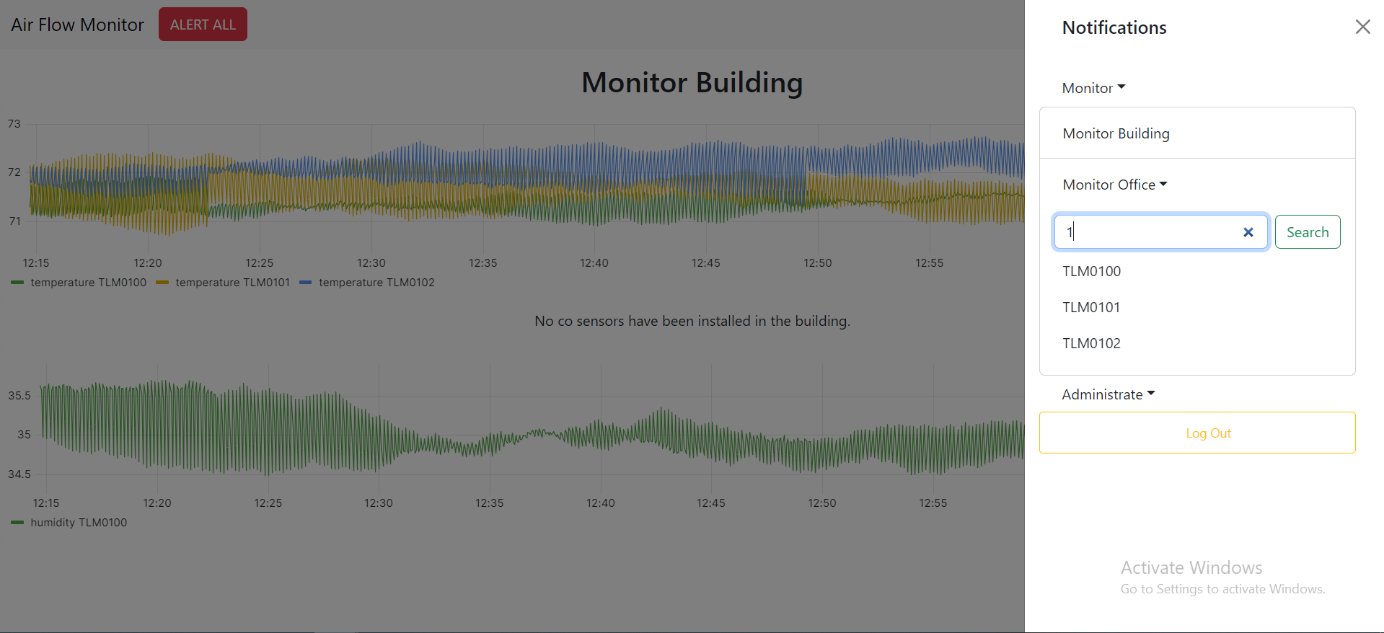
Након успјешне пријаве на систем, корисник ће бити преусмјерен на *Monitor Building* страницу. На овој страници, доступни су различити графикони који приказују вриједности очитавања са одређених типова сензора на нивоу цијеле зграде. На сваком графикону, подаци су представљени за све постављене сензоре одређеног типа, на примјер, температуре. У легенди графикона, имена канцеларија у којима су сензори постављени, омогућавају кориснику лакше идентификовање и анализу вриједности.

Изглед *Monitor Building* странице за кориснике типа радник одржавања система и администратор приказан је на слици 6.2.2.1. Док корисници типа администратор и радник одржавања система имају видљиво дугме „*Alert All“*, корисници типа радник неће видјети ово дугме на страници. Због ове мале разлике, слика изгледа странице за кориснике типа радник неће бити приложена.

*Слика 6.2.2.1 Изглед странице за преглед очитавања са свих сензора у згради*

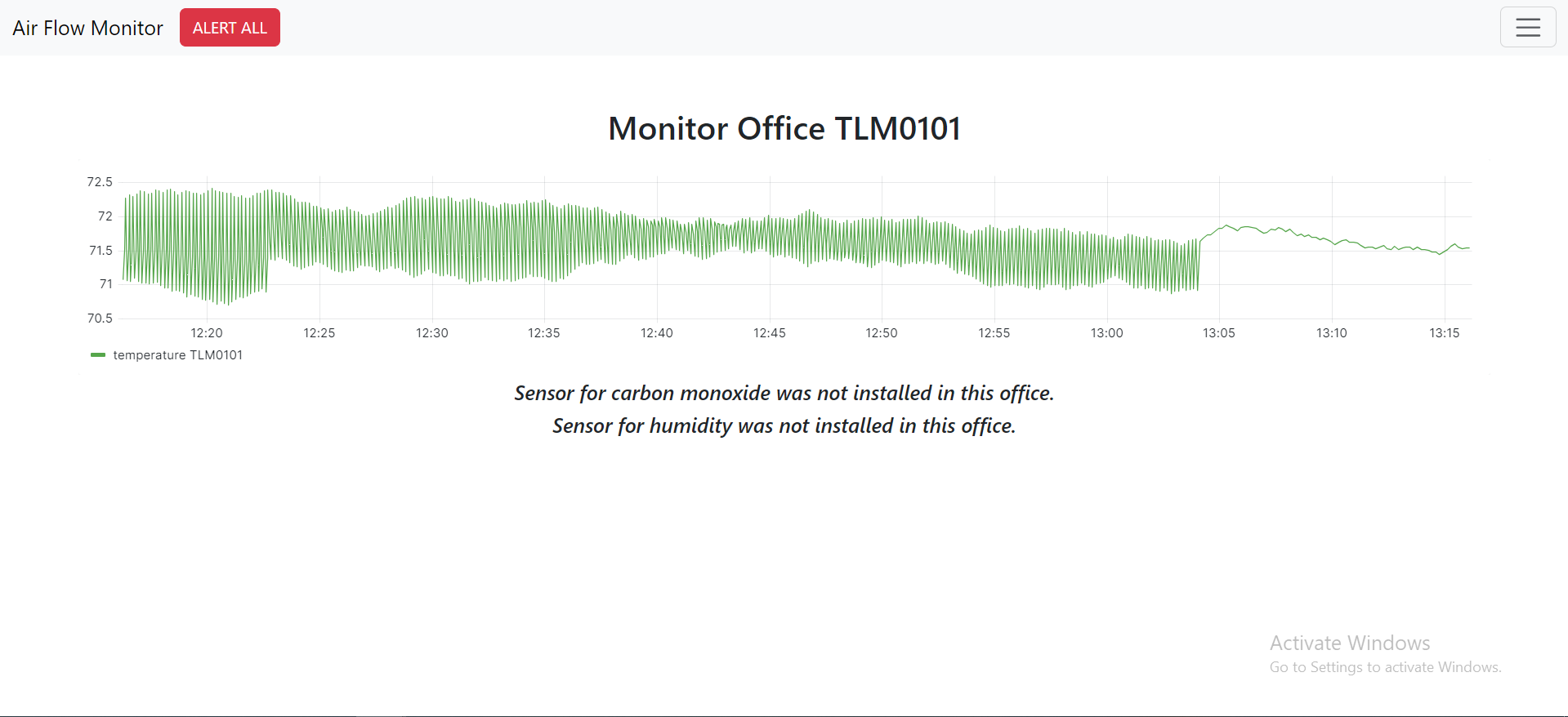
### Надгледање параметара на нивоу канцеларије

Са циљем ефикасног праћења података, систем омогућава корисницима да надгледају све сензоре на нивоу канцеларије. Путем навигационог *menu*-a, одабиром опције *Monitor Office*, добија се преглед свих регистрованих канцеларија у систему. Такође, постоји опција претраживања канцеларија по њиховом називу, као што је приказано на слици 6.2.2.2, што доприноси бржем и прецизнијем приступу жељеној локацији.



*Слика 6.2.2.2 Претрага канцеларије по називу*

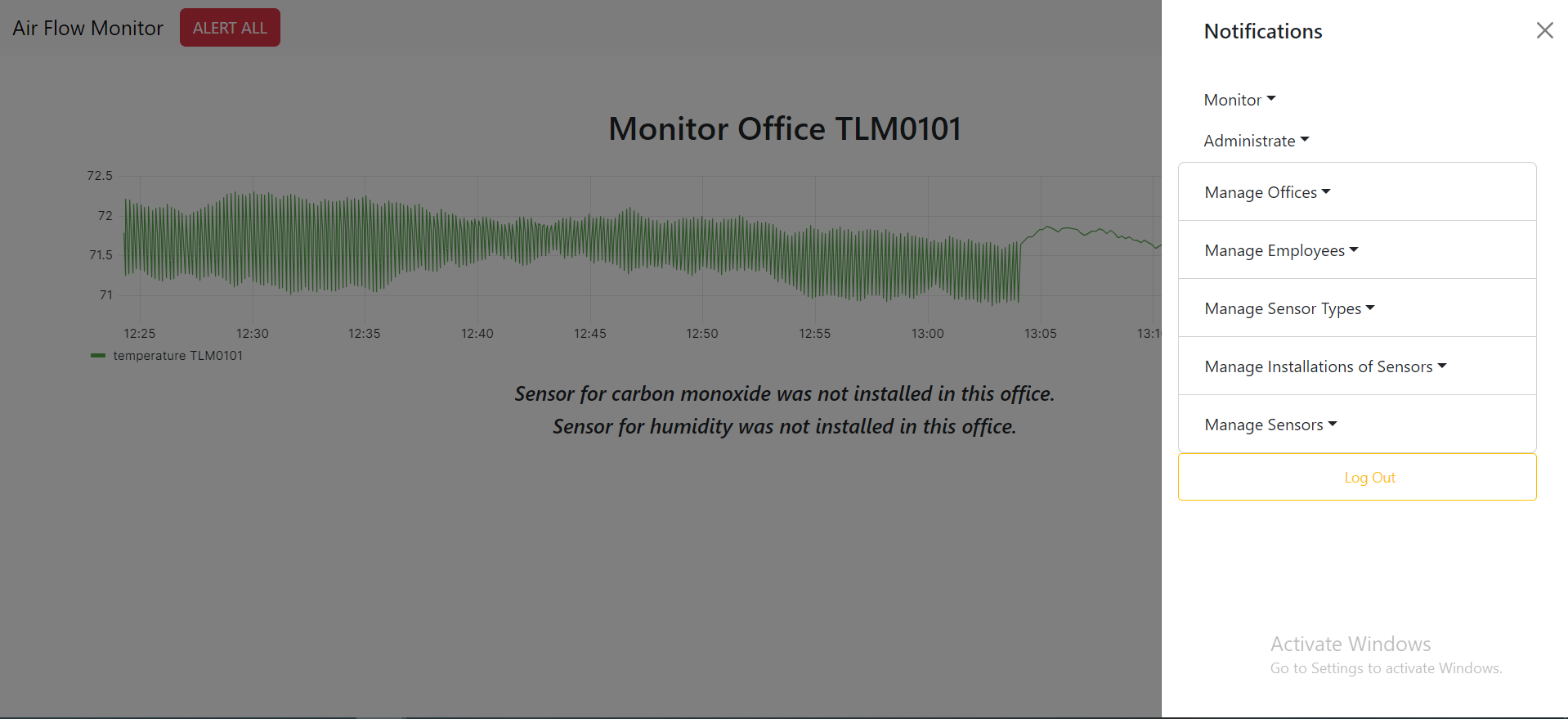
Након избора конкретне канцеларије, кориснику је омогућен преглед графикона за сваки сензор у том простору, као што је приказано на слици 6.2.2.3.



*Слика 6.2.2.3 Изглед странице за преглед очитавања са свих сензора у канцеларији*

### Приказ, креирање, брисање и модификовање канцеларија, радника, типова сензора, сензора и инсталација

Корисник типа администратор има привилегован приступ одређеном дијелу система, чија је функција евидентирање стања у пословној згради. Администратору је омогућен приступ овом дијелу система путем навигационог *menu*-а, конкретно избором опције *Administrate*. Важно је истаћи да корисници других типова немају овлаштење за приступ овом делу система.



*Слика 6.2.4.1 Приказ администраторског дијела навигационог menu-a*

У наставку ће бити приказан интерфејс за менаџмент инсталација сензора и канцеларија. Због сличности, странице за читање, креирање, брисање и модификовање осталих ентитета неће бити детаљно илустроване.

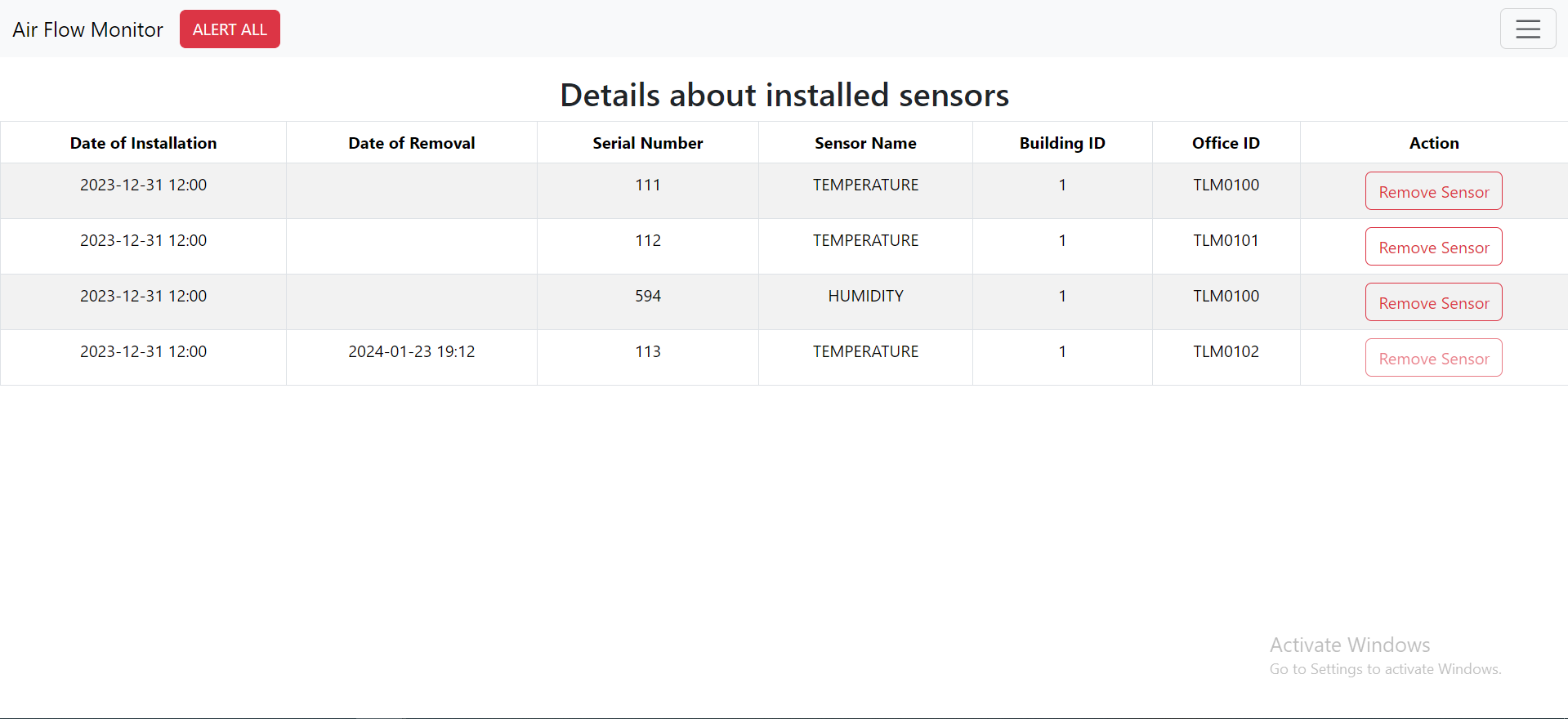
#### Менаџмент инсталација сензора

При регистрацији типа сензора у систему, као што је на примjер сензор температуре, кориснику је затим омогућено да евидентира инсталацију конкретног сензора, као што је сензор температуре са серијским бројем 111, у одређену канцеларију.

На слици 6.2.4.1.1, приказан је преглед свих извршених инсталација у систему. Корисник може приступити овом прозору тако што изабере опцију *Administrate* у *menu*-у, затим пређе на *Manage Installations of Sensors* и одабере *View Installations*.

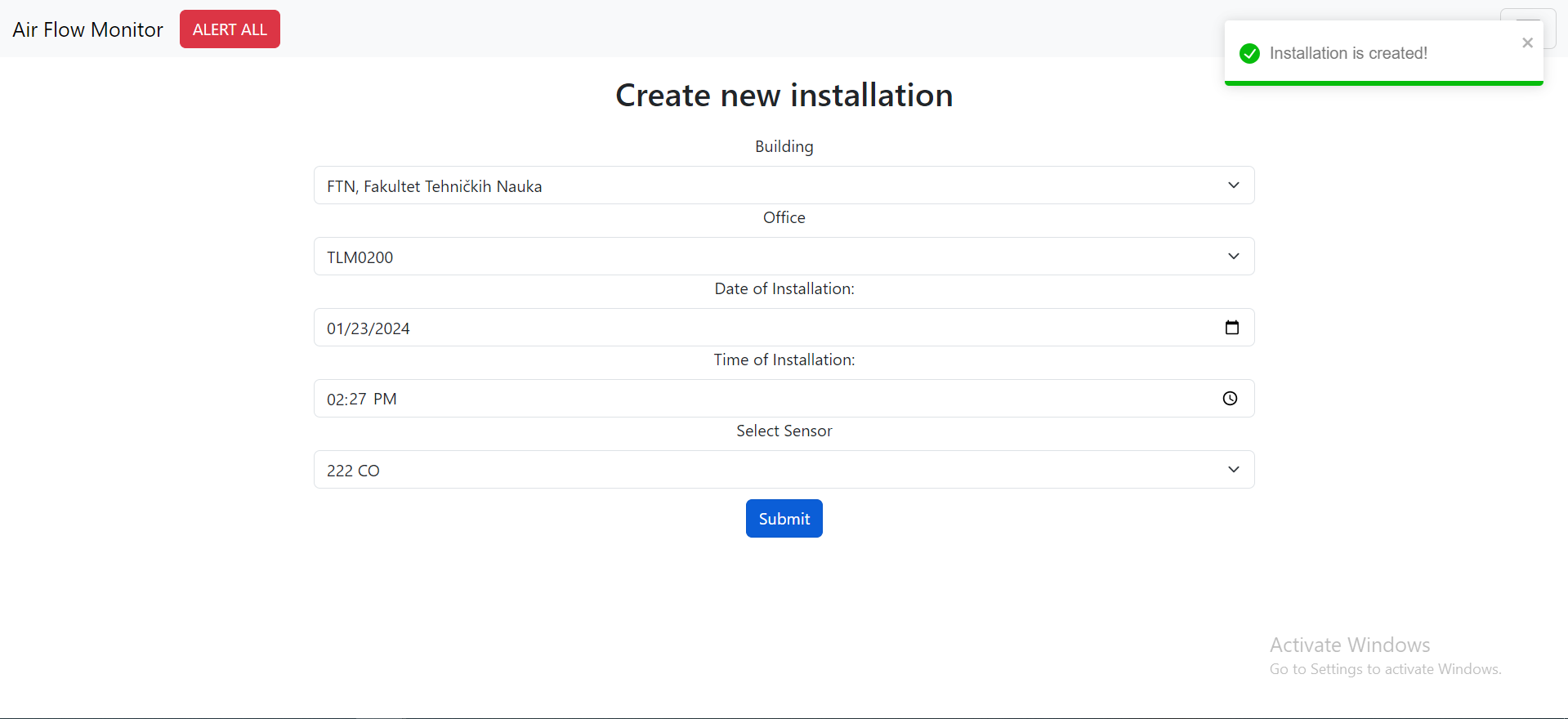
У табели су представљене значајне информације: датум и вријеме инсталације сензора, серијски број и тип инсталираног сензора, као и информације о згради и канцеларији гдје је сензор постављен.

Кликом на дугме „*Remove Sensor“*, корисник може евидентирати да је одређени сензор изашао из функције. У том случају, систем ће приказати датум и вријеме када је сензор стављен ван функције, а дугме „*Remove Sensor“* постаје онемогућено.



*Слика 6.2.4.1.1 Изглед странице за приказ свих инсталација*

Евидентирање нове инсталације сензора врши се на страници за креирање инсталација. Корисник може приступити овој страници тако што изабере опцију *Administrate* у *menu*-у, затим пређе на *Manage Installations of Sensors* и одабере *Create Installations*.

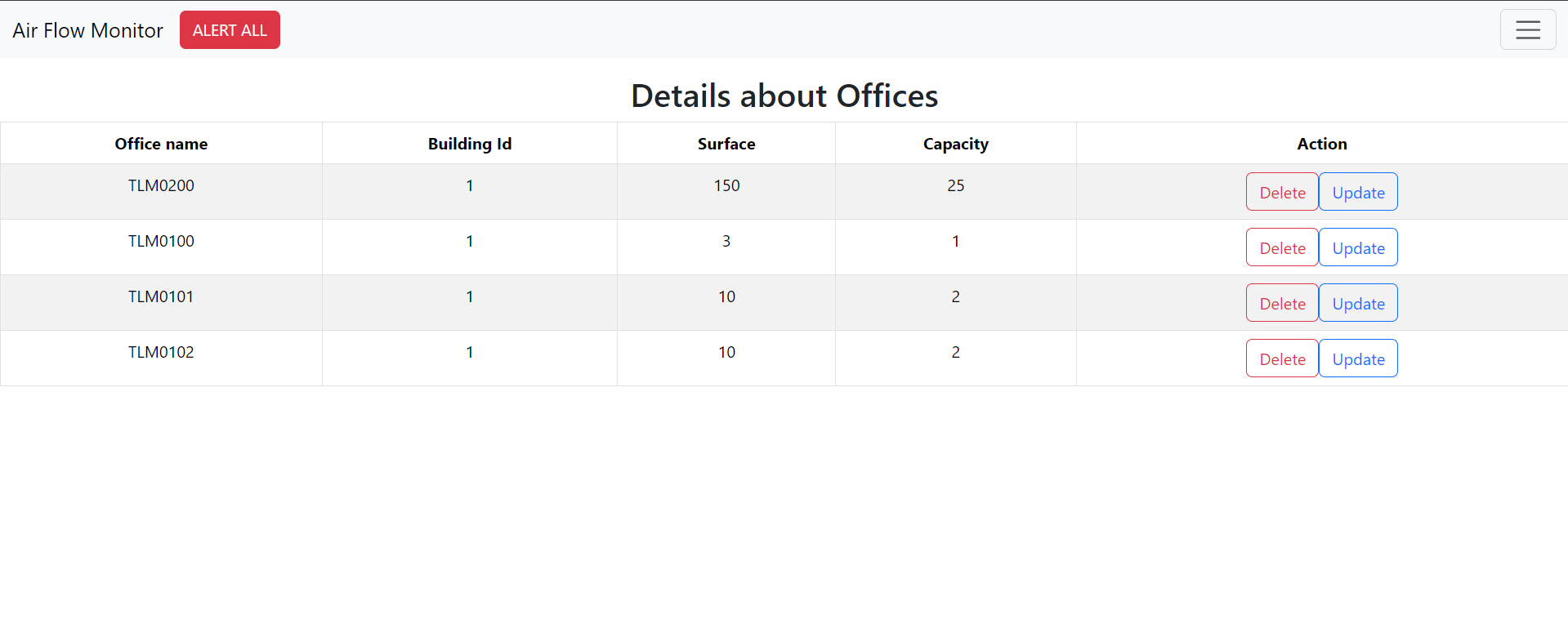


*Слика 6.2.4.1.2 Изглед странице за креирање инсталације*

У форми за креирање нове инсталације, кориснику је потребно да унесе информације о згради, канцеларији, датуму и времену инсталације, као и типу сензора који је постављен. Након успешног попуњавања ових поља, корисник извршава жељену акцију кликом на дугме „*Submit“,* као што је приказано на слици 6.2.4.1.2.

#### Менаџмент канцеларија

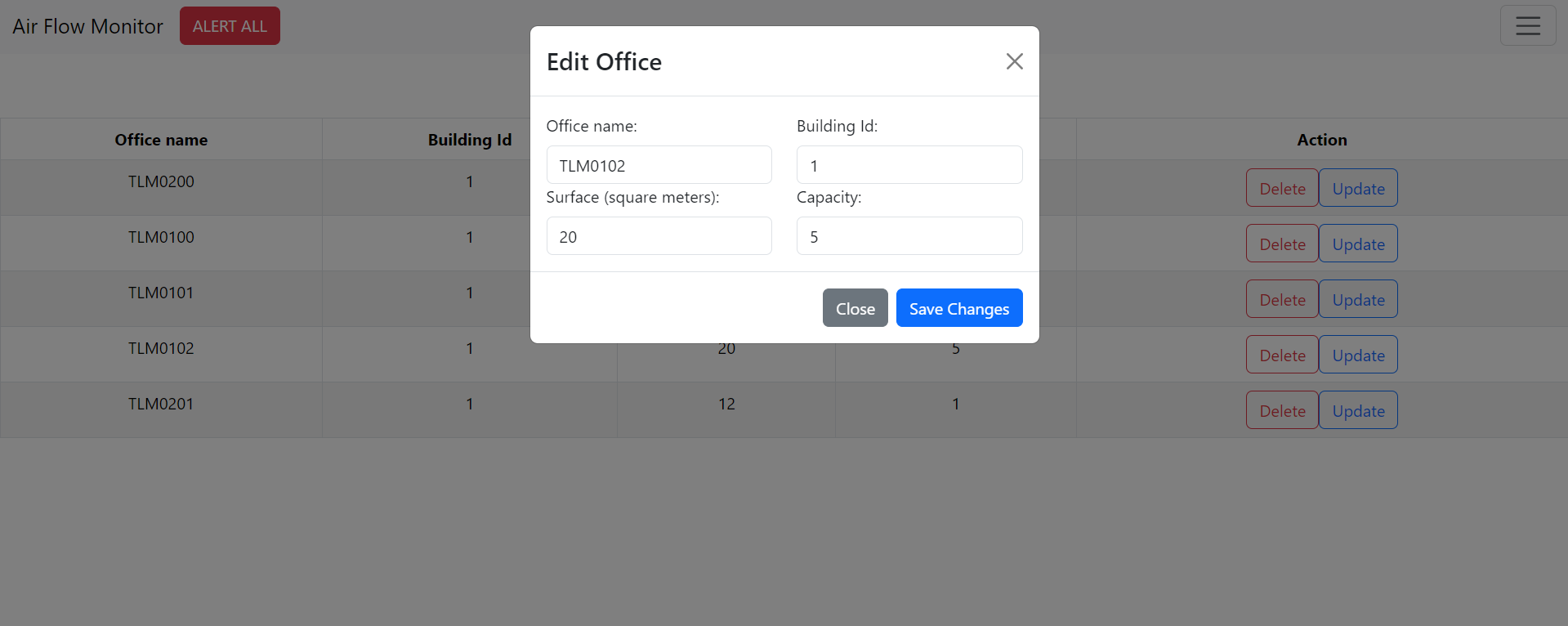
Слично као за инсталације сензора, могуће је управљати канцеларијама у систему. Подржане операције укључују преглед свих канцеларија, креирање нових, модификовање и брисање постојећих. Изглед странице за приказ свих канцеларија је представљен на страници *View Office,* приказаној на слици 6.2.4.2.1, до које се долази кроз мени *Administrate* -> *Manage Offices* -> *View Office.*



*Слика 6.2.4.2.1 Изглед странице за приказ свих канцеларија*

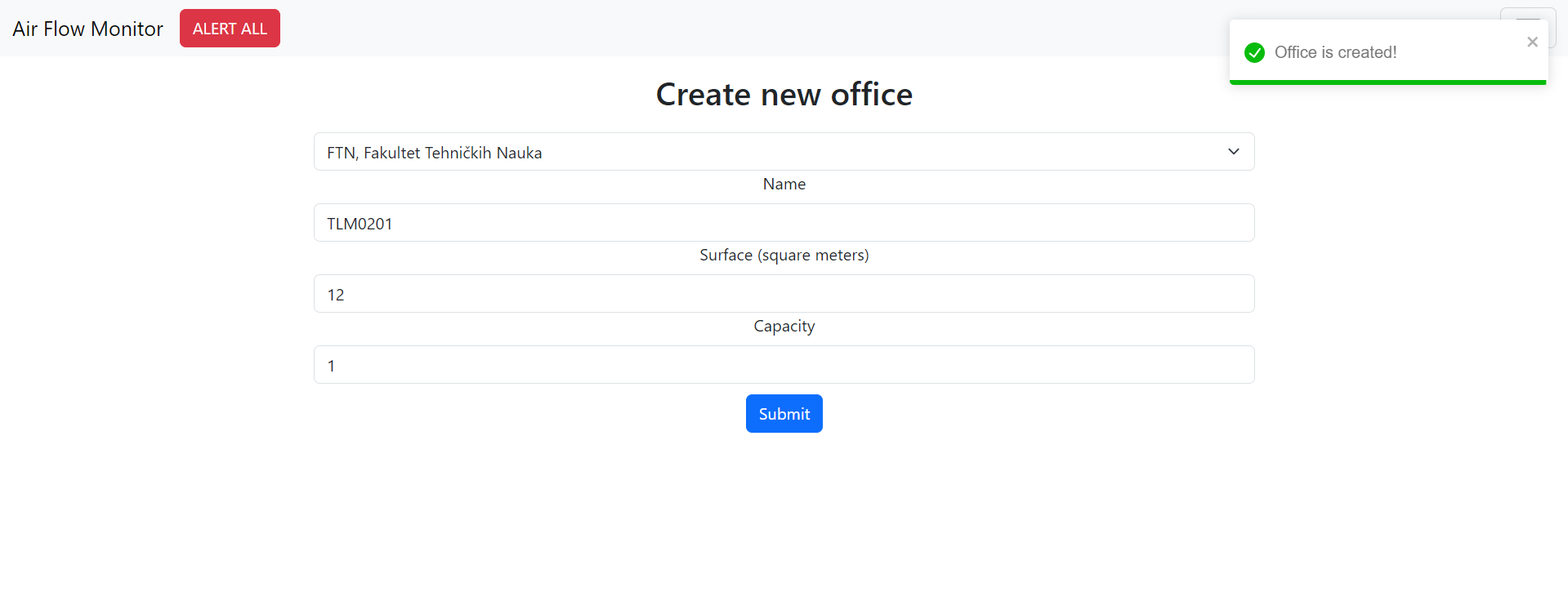
Канцеларији је могуће модификовати површину и капацитет радних мјеста у њој, кликом на дугме „*Update“,* након извршених промјена, нове вриједности је могуће сачувати кликом на дугме „*Save Changes*“ или одбацити кликом на дугме „*Close“*.

Изглед прозора за модификовање постојеће канцеларије је приказан на слици 6.2.4.2.2.

*Слика 6.2.4.2.2 Изглед прозора за модификовање постојеће канцеларије*

Кликом на дугме „*Delete“* у оквиру *View Office* странице могуће је обрисати канцеларију из система ако она нема у себи инсталиран нити један сензор.

Акције *Administrate -> Manage Offices-> Create Office* ће довести корисника на страницу *Create Office* илустровану на слици 6.2.4.2.3.

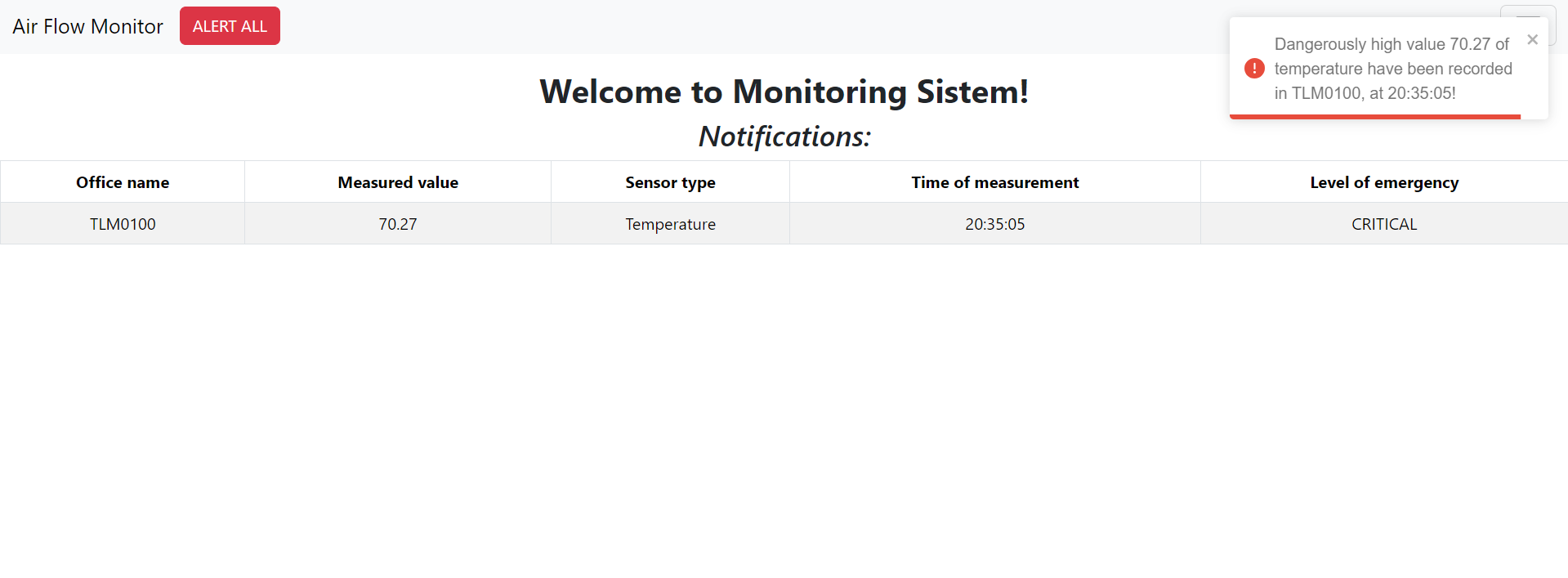


*Слика 6.2.4.2.3 Изглед странице за креирање нове канцеларије*

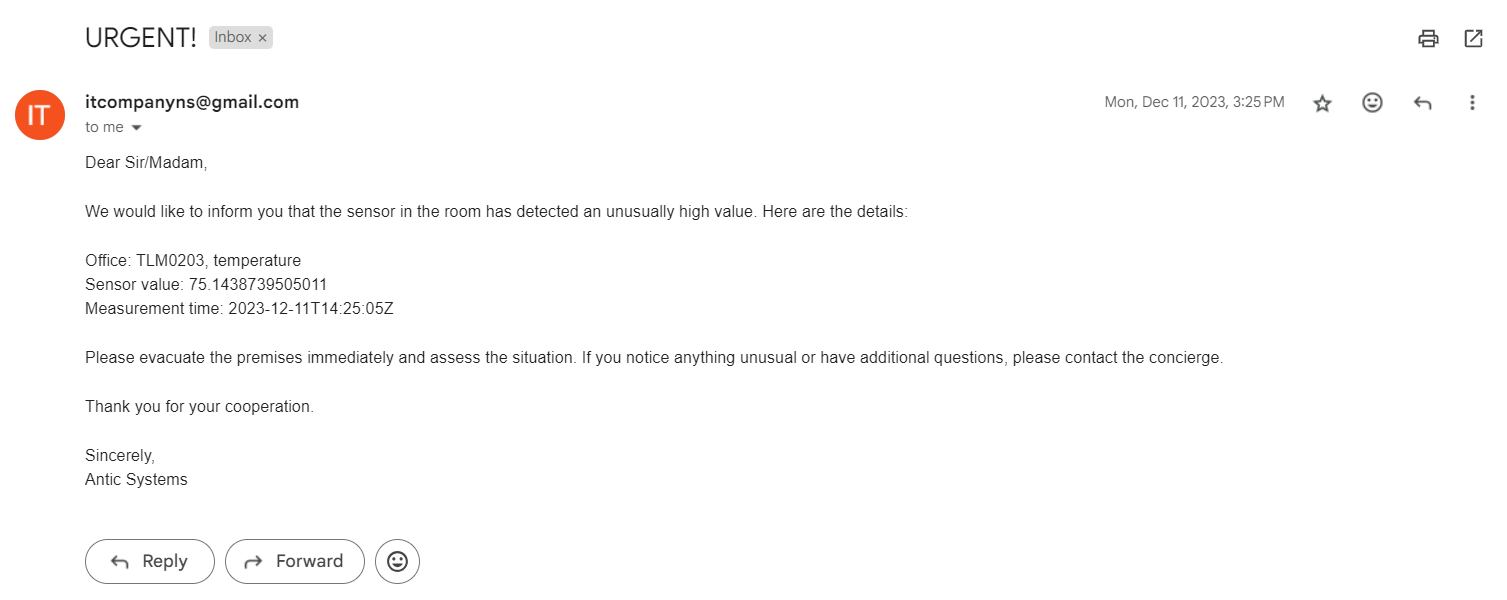
У форми за креирање нове канцеларије, кориснику је потребно да унесе информације о згради у којој се налази, називу, површини у квадратним метрима као и броју радних мјеста. Након успјешног попуњавања ових поља, корисник извршава жељену акцију кликом на дугме „*Submit“*, након чега ће му се приказати порука о успјешности акције.

#### Обавјештења

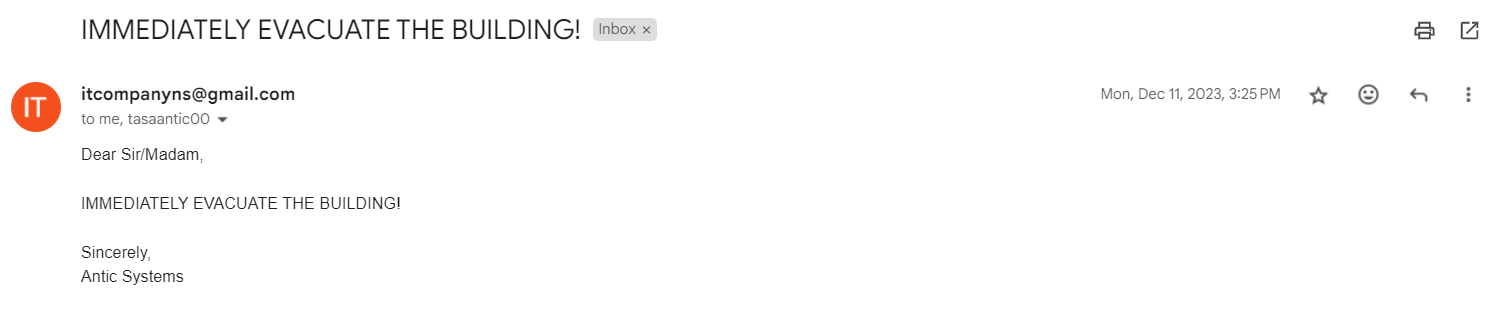
Кориснику се сва обавјештења приказују у облику *toast* порука у горњем десном дијелу екрана. Додатно, сва обавјештења биће приказана у табели на страници *Notifications*. Изглед те странице представљен је на слици 6.2.4.3.1.



*Слика 6.2.4.3.1 Изглед странице за обавјештења*

Кориснику ће одговарајућа обавјештења бити достављена путем електронске поште. Може да очекује два различита типа електронских порука – једна у формату упозорења, као што је приказано на слици 6.2.4.3.2, и друга у формату наредбе о евакуацији, као што је приказано на слици 6.2.4.3.3.

*Слика 6.2.4.3.2 Садржај мејла упозорења*

Корисници типа радник одржавања система и администратор имају могућност да уколико примијете необичне измјерене вриједности, кликну на дугме „*Alert All“*. Овом акцијом, мејлом ће бити упућено наређење свим запосленима тренутно присутним у згради, сугеришући да хитно напусте зграду, како је приказано на слици 6.2.4.3.3.

*Слика 6.2.4.3.3 Садржај мејла наређења о евакуацији*

#### Одјава

Корисник се може одјавити из система кликом на дугме „*Log Out“* у навигационом *menu*-у.

# Приједлози за даља усавршавања

Упркос томе што информациони систем већ пружа кориснику заокружен скуп функционалности, постоји простор за надоградњу и усавршавање. Узимајући у обзир предвиђену дуготрајну употребу система и очекивани експоненцијални раст броја корисника и података, потребно је истражити могућности надоградње безбједности, оптимизације базе и одржавања система. У наставку ће бити објашњени могући начини надоградње наведених аспеката система.

## Безбједност система

У свијету гдје je безбједност података од кључне важности, тренутна употреба нешифрованог *HTTP* протокола представља потенцијалну пријетњу интегритету података. Прелазак на *HTTPS*, енкриптовани протокол комуникације, не само да пружа заштиту од прислушкивања, већ и осигурава аутентичност комуникације између клијента и сервера. Ова надоградња би обезбиједила сигурну размјену података између крајњих корисника и система, минимизирајући ризик од потенцијалних напада и недозвољеног приступа подацима.

Aутентикација у тренутном систему ce ослањa на *JSON* токене, гдје само корисници са важећим токеном могу приступити систему. Тренутни систем имплементира ауторизацију на ниву клијентског дијела апликације, што подразумијева да видљивост одређених команди на корисничком интерфејсу зависи од улоге корисника у систему.

Оно што би додатно унаприједило систем јесте увођење концепта дозвола на нивоу серверског дијела, како би се спријечило потенцијално обилажење ауторизације на клијентском дијелу. Додавањем овог додатног слоја дозвола на серверској страни постигло би се додатно ојачавање безбједности система. Прецизна контрола приступа ресурсима осигурала би да само корисници са одговарајућим дозволама могу

извршавати одређене акције, неовисно о информацијама које су доступне на корисничком интерфејсу. Овај корак осигурава да чак и ако корисник има важећи токен, његов приступ одређеним дијеловима система биће додатно контролисан и ограничен према дефинисаним дозволама. Ова побољшања чине систем отпорнијим на потенцијалне безбједносне пријетње и смањују ризик од неовлаштеног приступа.

## Евиденција записа (енг. *log*)

Евиденција записа представља кључни алат у дијагностици и анализи перформанси система. Детаљан запис догађаја пружа увид у рад система, олакшава идентификацију потенцијалних проблема и омогућава брзу реакцију на неправилности. Континуирано праћење записа помаже у откривању шаблона понашања система, што даље доприноси побољшању његове стабилности и пoузданости. Интеграција система за евиденцију записа не само да подржава дијагностичке активности, већ представља и кључни инструмент у превентивном одржавању, чиме се смањује потенцијални утицај неправилности на корисничко искуство.

## Оптимизација базе података

Брзо прикупљање података, сваких 5 секунди у овом систему, значајно доприноси расту базе података. Како би се одржало ефикасно управљање подацима, могуће је извшавати скалирање података старијих од мјесец дана. Процес скалирања укључује израчунавање просjечних вриједности података у временском оквиру од пола сата и одбацивања остатка података, чиме се значајно смањује укупан волумен података. Овај приступ, осим што доприноси оптимизацији базе података, има и позитиван утицај на перформансе система. Ова стратегија не само да омогућава ефикасно управљање подацима већ и дугорочно осигурава одрживост система без губитка битних информација.

# Закључак

Овај дипломски рад представља успјешну имплементацију система за надгледање сензора у канцеларијским просторима, са фокусом на праћење температуре, влажности и нивоа угљен моноксида у реалном времену. Пројекат је осмишљен са циљем да обезбиједи ефикасну обраду и визуализацију великих количина података, омогућавајући корисницима брз и прецизан увид у околности у канцеларијским просторијама.

Кључне функционалности система, укључујући аутоматско обавјештавање о изласку очитаних вриједности са сензора из нормалних опсега и позиву за евакуацију при екстремним очитавањима, чине систем изузетно реактивним на потенцијалне опасности. Визуализација података путем графикона доприноси транспарентности и лаком праћењу параметара на нивоу зграде и канцеларија.

Потенцијално побољшање система може да се фокусира на неколико битних аспеката.

Оптимизација базе података путем скалирања старијих информација има за циљ подршку у ефикасном управљању подацима и одржавању оптималних перформанси система.

У контексту усавршавања дијагностичких и аналитичких могућности, интеграција система за евиденцију записа апликације не само да олакшава рјешавање потенцијалних проблема, већ такође доприноси одржавању стабилности цјелокупног система.

Побољшање безбједности може бити реализовано увођењем енкриптованог *HTTPS* протокола, што ће обезбиједити повећан степен аутентичности и заштиту од потенцијалних пријетњи.

Да ли је побољшање система неопходно и у којој мјери зависи од конкретне употребе система. Важно је размотрити природу и обим података, као и потребу за заштитом приватности и сигурности, што ће допринијети правилном одабиру стратегије за побољшање система, у складу са конкретним потребама и захтјевима корисника.

# Литература

1. *„Time Series Databases New Ways to Store and Access Data“* аутора *Ted Dunning* и *Ellen Friedman.* [На мрежи]. Доступно на:

<https://web.cs.wpi.edu/~cs585/s17/Books/Books-PDF/Time_Series_Databases.pdf>

1. „*InfluxDB Documentation*“. [На мрежи]. Доступно на:

<https://docs.influxdata.com/>

1. „*Grafana Documentation*“. [На мрежи]. Доступно на:

<https://grafana.com/docs/grafana/latest/>

# Списак кориштених скраћеница

***API*** *- Application Programming Interface*

***int64*** *– integer, 64 bit*

***float64*** *– float, 64 bit*

***SQL*** *- Structured Query Language*

***NoSQL*** *-**Not Only SQL*

***ACID*** *- Atomicity, Consistency, Isolation, Durability*

***HTTP -*** *Hypertext Transfer Protocol*

***HTTPS -*** *Hypertext Transfer Protocol Secure*

***URL -*** *Uniform Resource Locator*

# Биографија

Тамара Антић рођена је 23. маја 2000. године у Бањој Луци. Завршила је Гимназију у Баљој Луци, општи смјер. Школске 2019/2020. године уписао је Факултет Техничких Наука у Новом Саду, одсек Рачунарство и аутоматика, усмерење Примењене рачунарске науке и информатика. Положила је све испите предвиђене планом и програмом.