Heat Production Management Project for Semester Project 2

Kacper Grzyb Sebestyen Deak Ignad Bozhinov
Leonardo Gianola Levente Sohar

03-06-2024

Contents

Chapter 1

Introduction

Introduction chapter goes here

Chapter 2

Release Planning

Release Planning chapter goes here

Chapter 3

Sprint Materials

Sprint Planning Chapter goes here

Chapter 4

Technical Details

Technical Details Chapter goes here

4a Software Architecture and Design

Tools

The team decided to use the ASP.NET Core framework's Razor Pages for building the project for reasons such as:

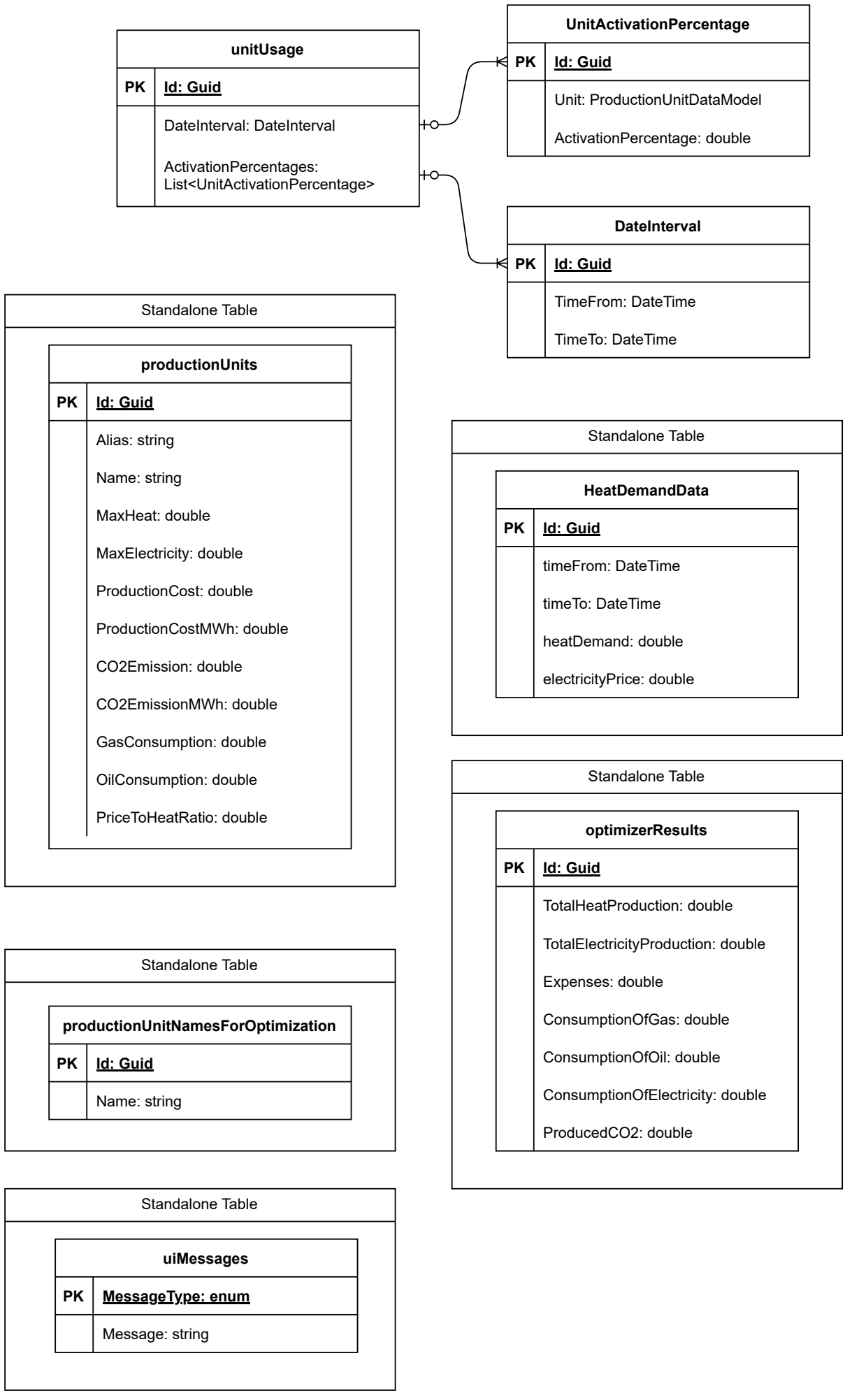
- The most important argument for using ASP.NET Core is it's cross-platform functionality. The developers in the team use both Macintosh and Windows based systems, therefore a framework that could switch seamlessly between them was crucial.
- As the name suggests, the framework runs in the .NET ecosystem, which is what the team has been taught in the course so far therefore it is what the team is most experienced and most comfortable working in.
- With Razor Pages being a web-page based solution, the UI is mostly composed of HTML and CSS, which some team members already had experience in and the rest was eager to learn. A light-weight, web-based solution allowed the team to be more flexible, and develop the app at a more rapid pace compared to if the team chose a Model-View-Controler (or a Model-View-ViewModel) solution. This freedom allowed for more features and better adjustment to changing the project requirements.

As for other tools, the team used:

- Github: Source and Version Control, Collaborative Development of the App
- Jira: Task Management and Planning as well as adhearing to Agile which was one of the requirements for the project
- Discord: Communication and Resource Sharing
- diagrams.net: Creating UML Diagrams for this chapter
- Figma: Prototyping and General UI Design
- Visual Studio and Visual Studio Code: Code Editors

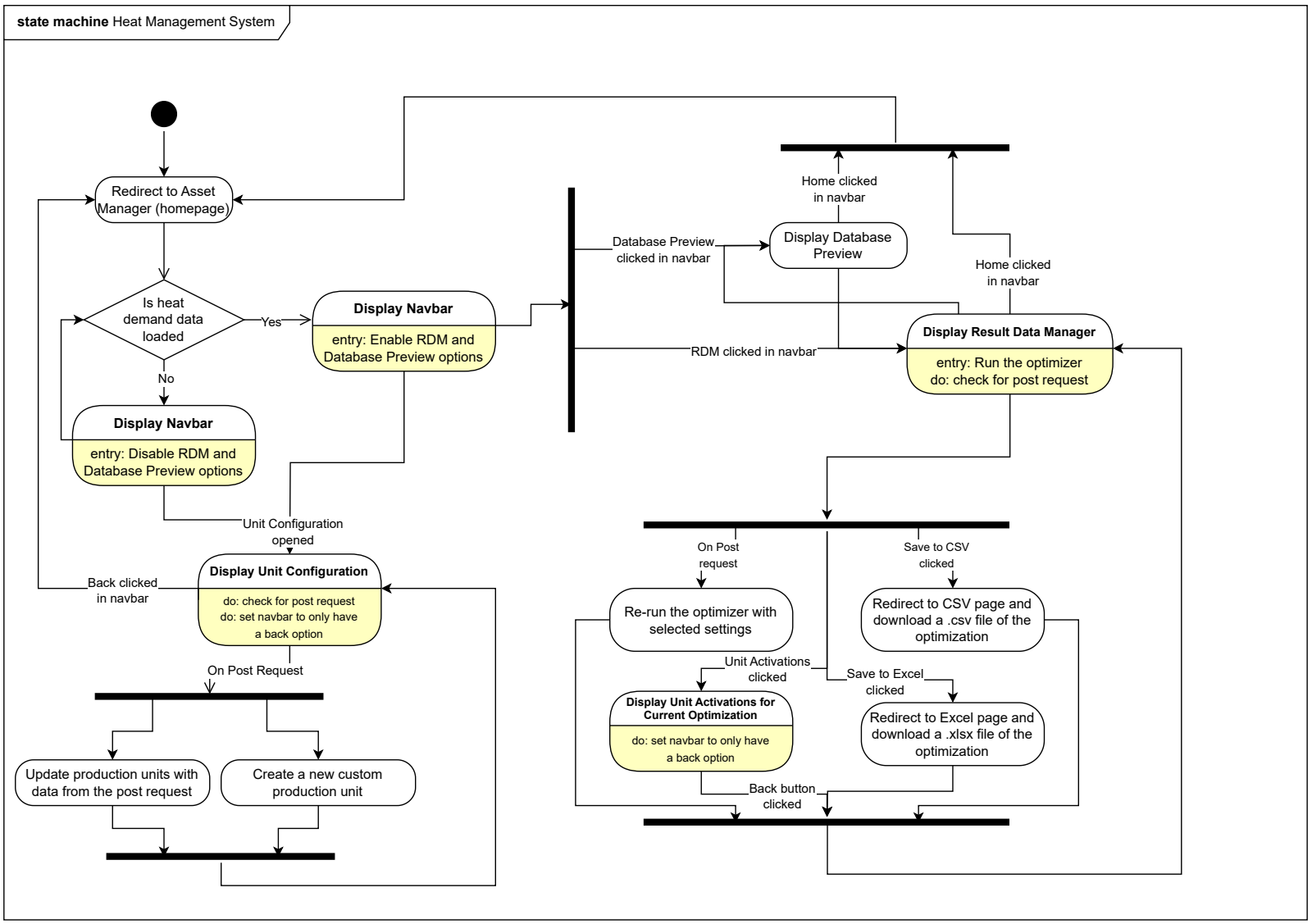
Database Architecture

Before diving into the program architecture the in-memory database solution offered by Razor Pages must first be mentioned. In order to achieve data persistance while switching in between pages in a Razor Pages project, a database must be used. Since we did not want to go too far out of the scope of the project, we decided not to use a dedicated database solution like a MySQL or MSSQL Server for this project, expecially because the team has not had any database modeling courses yet. Instead we chose a middle-ground, which is the before mentioned in-memory database. This soultion offers similar functionality to a real database, with the comfort of running in the program's memory, which eliminates potential connection, authentication privilege and/or security risks and issues connected with using databases.


unitUsage
PK
Id: Guid
DateInterval: DateInterval
ActivationPercentages: List<UnitActivationPercentage>
UnitActivationPercentage
PK
Id: Guid
Unit: ProductionUnitDataModel
ActivationPercentage: double
DateInterval
PK
Id: Guid
TimeFrom: DateTime
TimeTo: DateTime
Standalone Table
productionUnits
PK
Id: Guid
Alias: string
Name: string
MaxHeat: double
MaxElectricity: double
ProductionCost: double
ProductionCostMWh: double
CO2Emission: double
CO2EmissionMWh: double
GasConsumption: double
OilConsumption: double
PriceToHeatRatio: double
Standalone Table
HeatDemandData
PK
Id: Guid
timeFrom: DateTime
timeTo: DateTime
heatDemand: double
electricityPrice: double
Standalone Table
optimizerResults
PK
Id: Guid
TotalHeatProduction: double
TotalElectricityProduction: double
Expenses: double
ConsumptionOfGas: double
ConsumptionOfOil: double
ConsumptionOfElectricity: double
ProducedCO2: double
Standalone Table
productionUnitNamesForOptimization
PK
Id: Guid
Name: string
Standalone Table
uiMessages
PK
MessageType: enum
Message: string

The database does not fully comply to standard database structures one could see when dealing with real database systems. We do see the limitations of this design and the flaws inside of it, such as the lack of connections between tables and the lack of foreign keys inside of each table. Instead some fields in the tables use custom class data types for ease of use. Despite the flaws of using DbSets, we found that their functionality was sufficient for the scope of the project. It is also worth to mention, that due to our lack of experience with Razor Pages, we were unsure of how DbSets and in-memory data structures function. Because of that in the middle of development the database had to be refactored from a more json-like data storage structure to one that complies with standards enforced by DbSets, such as using Primary Keys. It is because of this process that we ended up having a mixture of both structures.

Applicaion Flow


state machine Heat Management System
Redirect to Asset Manager (homepage)
Is heat demand data loaded
Yes
No
Display Navbar
entry: Enable RDM and Database Preview options
Display Navbar
entry: Disable RDM and Database Preview options
Unit Configuration opened
Display Unit Configuration
do: check for post request
do: set navbar to only have a back option
Back clicked in navbar
On Post Request
Update production units with data from the post request
Create a new custom production unit
Home clicked in navbar
Database Preview clicked in navbar
Display Database Preview
Home clicked in navbar
RDM clicked in navbar
Display Result Data Manager
entry: Run the optimizer
do: check for post request
On Post request
Re-run the optimizer with selected settings
Unit Activations clicked
Display Unit Activations for Current Optimization
do: set navbar to only have a back option
Save to CSV clicked
Redirect to CSV page and download a .csv file of the optimization
Save to Excel clicked
Redirect to Excel page and download a .xlsx file of the optimization
Back button clicked

4b Simple Design

Simple design yapping goes here

4c Incremental Design

Incremental Design yapping goes here

4d Refactoring

Refactoring yapping goes here

4e Test-Driven Development

Test-Driven Development yapping goes here

4f Unit Testing

Unit Testing yapping goes here

4g Pair Programming

Pair Programming yapping goes here

4h Code Review

Code Review yapping goes here

Chapter 5

Conclusion and Group's Reflections

Conclusion chapter goes here

5a Working on a common project with other groups

5a yapping goes here

5b What went well and not so well with the group's specific set of tasks

5b yapping goes here

5c Specific contributions of each team member

5c yapping goes here

5d Future actions to prevent problems and difficulties faced during the project

5d yapping goes here