TODO: Headhight noch zu 12pt zurückstellen

htblakaindorf
Leistung mit Menschlichkeit

informatik

Höhere Technische Bundeslehranstalt Kaindorf an der Sulm

Abteilung Informatik

Diplomarbeit

im Rahmen der Reife- und Diplomprüfung

Königskarte

informatik

Leon Edlinger
Paul Gigler
Andreas Weissl

5BHIF
2024/2025

Betreuer: Prof. DI Johannes Loibner, BSc
Projektpartner: Prof. DI Robert Müllerferli
Datum: MISSING DATE

Statutory declaration

I declare under oath that I have written the present diploma thesis independently and without outside help, have not used sources and aids other than those indicated and have identified the passages taken from the sources used literally and in terms of content as such.

Ort, Datum	Leon Edlinger
Ort, Datum	Paul Gigler
Ort, Datum	Andreas Weissl

Abstract

Abstract in English

Kurzfassung

Kurzfassung in Deutsch

Thanks

It would not have been possible to carry out this thesis to this extent without the active support of a number of people. We would therefore like to thank everyone who supported us in the implementation of this thesis.
…

Table of Contents

1 Introduction **1**
1.1 Team . . . 2
1.2 Motivation . . . 2

2 Technologies **3**
2.1 Frontend . . . 3
2.1.1 Dart . . . 3
2.1.2 Flutter . . . 3
2.2 Backend . . . 4
2.2.1 Java Spring . . . 4
2.2.2 PostgreSQL . . . 4
2.3 Version Control . . . 5
2.3.1 Git . . . 5
2.3.2 GitHub . . . 5
2.4 Map Data . . . 6
2.4.1 OpenStreetMap . . . 6
2.4.2 GraphHopper . . . 6
2.5 Development Tools . . . 7
2.5.1 VS Code . . . 7
2.5.2 IntelliJ . . . 7
2.5.3 Android Studio . . . 7
2.5.4 Postman . . . 8
2.6 Deployment . . . 9
2.6.1 Docker . . . 9

3 Wireframes **10**
3.1 Admin Ansicht . . . 10
3.2 User Ansicht . . . 10

4 Research Questions **10**
4.1 Leon Edlinger . . . 10
4.2 Paul Gigler . . . 10
4.3 Andreas Weissl . . . 10

5 Spring Framework **10**
5.1 Spring Boot . . . 10
5.2 Spring Data JPA . . . 10
5.3 Lombok . . . 10
5.4 Advantages . . . 10

6 Structure of the Backend **10**
6.1 Controller Layer . . . 11
6.2 Service Layer . . . 11
6.3 Repository Layer . . . 11
6.4 Persistence Layer (Entity Classes) . . . 11
6.5 Applied Design Principles (DTOs) . . . 11

7 Area Borders **11**
7.1 Purpose of Area Borders in the App . . . 11
7.2 Overview of the Convex Hull Algorithm . . . 11
7.3 Use Cases of the Convex Hull in Industry . . . 11
7.4 Alternate Methods for Area Border Calculation . . . 12
7.5 Rationale for Choosing the Convex Hull Method . . . 12
7.6 Integration of the Algorithm into the Backend . . . 12
7.7 Challenges and Adjustments . . . 12

8 Implementation **12**
8.1 Config of Spring Boot (application.properties) . . . 12
8.2 Entity Classes (Structure/Purpose) . . . 12
8.3 JpaRepositories (DB Access and CRUD Operations) . . . 12
8.4 Service Classes . . . 12
8.5 Rest Controller (API Endpoints and their Functions) . . . 12

9 Final Thoughts **13**
9.1 Leon Edlinger . . . 13
9.2 Paul Gigler . . . 13
9.3 Andreas Weissl . . . 13

10 Meetings **14**

11 Working Hours **15**

12 Source code directory **16**

13 List of figures **17**

14 List of tables **18**

15 Bibliography **19**

16 Abbreviation **20**

1 Introduction

TODO: Bitte noch anpassen falls euch der Wortlaut net so passt

Mobile Apps get used for quite literally everything in today's World. So after we noticed that there is no application that allows the efficient planning of campaigns like the "Sternsinger-Aktion" we asked ourselves why, and furthermore, how hard it would be, to create an App with intuitive Usability for the sole purpose of simplifying the process of managing such a campaign and gaining a general overview of the progress made.

TODO: Vielleicht noch kürzen, wirklich nur die groben Themen?

The research part of this thesis will be dedicated to how components should act and look, so that new users can use this tool without requiring a long "onboarding" phase. It should feel familiar to interact with elements and the borders of what users can and can't do need to be clearly defined. Because our application also needs a somewhat reliable data source to guarantee the consistency and accuracy of marked addresses in our app. For this purpose we researched ways to keep our database up-to-date, without the need of much manual intervention. After defining the projects requirements, we noticed that we need to somehow calculate which addresses are "Border" addresses. So we decided to take a look into different algorithms for this task and compare them concerning their efficiency and then decide on one of them and implement it.

This thesis contains an in-depth description of our thought and development process, as well as any other steps we took to achieve our goal of a functional mobile application that can be used by volunteers in course of the "Sternsinger-Aktion 2025" taking place in the parish of Lieboch.

TODO: Wortwiederholung austauschen

The result of this thesis should be a mobile app that provides users with the addresses that they need to visit on this day. They then should be able to easily mark the houses they already visited. If something unusual happens at this address, the user should be able to take note of this, so the organizers have knowledge of it and can account to it in the following year.

TODO: Maybe auf verschiedene Parts aufteilen, also das man zuerst sagt Problemstellung, dann Zielsetzung damit die Introduction übersichtlicher ist

1.1 Team

This thesis was created by three Students attending the BHIF20 at the HTBLA Kaindorf Computer Science Department.

TODO: andis bild anpassen

Leon Edlinger

Database, Admin-Panel

Paul Gigler

Deployment, Mobile App

Andreas Weissl

Backend

1.2 Motivation

2 Technologies

Development would not have been possible to implement without making use of many tools, frameworks and environments. In this chapter each tool used in the creation of our software will be described briefly.

2.1 Frontend

2.1.1 Dart

Dart is a programming language initially designed for web development, with the goal of replacing JavaScript, in mind. Today, it is used in a variety of software products, mainly because of the flutter framework. Dart can be compiled for many platforms and architectures, including ARM, x64, RISC-V, JavaScript or WebAssembly and is highly popular for its combination of high-level features, combined with practical language features like garbage collection and optional type annotation. It was developed by Google and is now an open-source project. [4]

Fig. 1: Dart Logo (Source: `https://dart.dev`)

2.1.2 Flutter

Flutter is an open-source software development framework. It allows programmers to compile their applications for different platforms including Web, macOS, iOS as well as Windows and any type of Linux-based systems, all from a single codebase, written in Dart. This allows for more efficient and faster cross-platform development. Another benefit of Google's toolkit is the highly customizable, predefined UI components. Developers can mix and match these components as needed which makes them an applicable choice. [5][3]
We chose flutter mainly for these reasons, but also because of our previous experience with Java to which Dart is quite similar. Through it, we were able to get started quickly, learn what we need along the way. Having a design through the components was also very helpful and saved us some time.

Fig. 2: Flutter Logo (Source: `https://flutter.dev/`)

2.2 Backend

2.2.1 Java Spring

Java on its own is an object-oriented programming language which is designed to be platform-independent. It allows programs to run on any device that has a Java Virtual Machine installed. Java is particularly useful for its strong security features and extensive community support.

Fig. 3: Java Logo (Source: `https://logodownload.org/java-logo/`)

Java Spring is a open-source framework and is based on Java. The Java Spring Framework is an in-depth programming and configuration language that simplifies development of Java applications. The most used Spring Framework is Spring Boot. Spring Boot simplifies the application process with not much need for preconfigured logic. Other Spring Frameworks which have a huge impact on Java programming are Spring Security for authentication/authorization and Spring Data for database injection. We chose Java Spring as our backend development tool, because of its simplicity and flexibility. As a framework, Spring allowed us to develop a well scalable and maintainable backend to generate APIs and connect to the database.

2.2.2 PostgreSQL

PostgreSQL is an open-source relational database management system. It is known for its reliability and scalability. It complies with SQL standard. Databases, which use PostgreSQL, are designed to handle a huge amount of data efficiently. Those databases support more advanced features such as full-text search and JSON data storage.

One of PostgreSQL's strengths is its flexibility. Developers are able to create custom data types and functions to customize the database to their specific needs. It also provides security mechanisms which include role-based access control and encryption.

We used PostgreSQL in our project as our database due to its ability to handle complex queries and its scalability which then allowed us to manage and store all the data reliably.

2.3 Version Control

2.3.1 Git

Git is a distributed VCS that was developed by Linus Torvalds in 2005. The main benefit of a VCS is to easily keep track of different versions of files. Git is the most widely used option, concerning this area of application. [12] This is due to the fact it is open-source, therefore free, as well as reliable and easy to learn. It gets used in almost every, but not only, development project, not just for tracking the history of files, but also for developing cooperatively making use of it's branching feature. This allows for development while maintaining a stable version on the main branch. [6]

Fig. 4: Git Logo (Source: `https://git-scm.com/`)

2.3.2 GitHub

GitHub is a platform maintained by Microsoft. As the Name implies, it is based on Git and provides the opportunity to easily share your Git repositories with other users. It is free to use for non-commercial applications and a very popular option when it comes to developing in a team. [1] Furthermore, it also implements handy extensions that integrate tightly with Git, for example GitHub-Actions, which is their solution for CI/CD Pipelines. There are many alternatives to GitHub, like, GitLab, which is open-source and can be self-hosted or BitBucket, a solution by Atlassian. [11]

Fig. 5: GitHub Logo (Source: `https://github.com/`)

2.4 Map Data

2.4.1 OpenStreetMap

The project OpenStreetMap (OSM) is an open-source initiative that aims to make high quality map data available to everyone for free. It was created in 2004 by Steve Coast. It was later widely adopted, with the main push being the pricing Google introduced to the embedding of Google Maps, so that today it can be found in all kinds of applications. Reaching from simple embedded web-components, all the way into video games like the Microsoft Flight Simulator, which relies on OSM for it building models. The GIS behind OSM gets maintained by volunteers and is feed not only using GPS-Data but also image data and other different formats.
OSM is the backbone of this thesis and without it, it wouldn't have been possible to develop this tool. [10] [16]

Fig. 6: OSM Logo (Source: `https://wiki.openstreetmap.org/`)

2.4.2 GraphHopper

GraphHopper (GH) in its core is an open-source routing engine written in Java. It can be deployed as a web server and is used to calculate the optimal route, distance or time between multiple different points. It supports multiple modes of transportation as well as "snap to road" technology. [7] GH also provides a ready to go API that you can access for free, but we choose to deploy our own instance since the requests we needed to make exceed the limits of the free version. [8]

Fig. 7: GraphHopper Logo (Source: `https://brandfetch.com/graphhopper.com`)

2.5 Development Tools

2.5.1 VS Code

Visual Studio Code (VS Code) is a version of the Code-OSS project with Microsoft-specific customizations, making it a closed-source product licensed by Microsoft. VS Code is highly customizable, with support for community-made plugins and many options for the user to define how things should act and look. [14] There are many handy features implemented out of the box, such as native VCS support. Since VS Code is an electron based application, it can not only be used on a desktop computer, but also directly in the browser of any device. [15] [13]

Fig. 8: VS Code Logo (Source: `https://code.visualstudio.com/brand`)

2.5.2 IntelliJ

IntelliJ IDEA is an IDE primarily designed for Java and Kotlin developers, published by JetBrains. In comparison to VS Code, it is more resource-intensive due to included tools like the direct integration of a database connection via the embedded DataGrip version. IntelliJ is the base framework used for other, more specific IDEs by JetBrains, like Android Studio, WebStorm or DataGrip. Its functionality can also be extended through plugins. [9] [2]

Fig. 9: IntelliJ IDEA Logo (Source: `https://www.jetbrains.com/company/brand/`)

2.5.3 Android Studio

Android Studio is based on the IntelliJ framework, but, as the name suggests, with the specification to developing Android apps. Aside from Java and Kotlin based pure Android apps, it can also be used for developing cross-platform applications using the flutter framework. The IDE introduced by Google also features an Emulator for Pixel devices which simplifies testing. Lastly, it provides an intuitive way to compile the app and flash it to a physical Android device using ADB and USB-debugging.

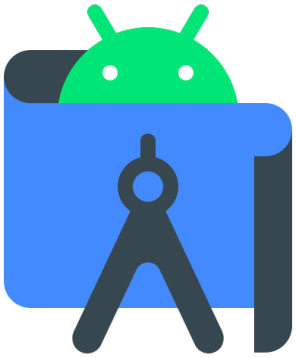

Fig. 10: Android Studio Logo (Source: https://img.icons8.com/?size=100&id=04OFrkjznvcd&format=png&color=000000)

2.5.4 Postman

Postman is an API platform, primarily used to test APIs. It features collaboration and an automatic code-generation, allowing users to design requests directly in postman and export them into any of the supported languages. We used it to develop and test our API, without a frontend application to minimize potential points of failure.

Fig. 11: Postman Logo (Source: https://www.postman.com/legal/logo-usage/)

2.6 Deployment

2.6.1 Docker

Docker is a tool for containerization and one of the most popular options in this field. Through containerization, it is possible to isolate a process from the host machine, in a virtual environment. Furthermore, docker is portable which makes it a great choice for software projects that many developers with different environments work on. It is used to create a controlled infrastructure which guarantees the correct configuration for the production server. We used Docker to host our PostgreSQL database across multiple different systems throughout the development process.

Fig. 12: Docker Logo (Source: `https://www.docker.com/`)

3 Wireframes

3.1 Admin Ansicht

3.2 User Ansicht

4 Research Questions

4.1 Leon Edlinger

4.2 Paul Gigler

4.3 Andreas Weissl

5 Spring Framework

The backend leverages the **Spring Framework**, a comprehensive framework for enterprise Java development. This section explores its key components and advantages.

5.1 Spring Boot

Spring Boot simplifies configuration and deployment with embedded servers and opinionated setups. This reduces boilerplate code and accelerates development.

5.2 Spring Data JPA

Spring Data JPA provides abstractions for database interactions, streamlining CRUD operations and custom query creation.

5.3 Lombok

Lombok reduces boilerplate code by generating getters, setters, and other methods at compile time, improving code readability and maintainability.

5.4 Advantages

Using Spring enhances productivity, reduces setup complexity, and ensures scalability, making it ideal for this project.

6 Structure of the Backend

The backend follows a layered architecture to promote separation of concerns, scalability, and maintainability. This section outlines the roles of each layer.

6.1 Controller Layer

The controller layer acts as the interface for incoming HTTP requests, delegating them to appropriate service methods.

6.2 Service Layer

The service layer contains business logic, validating data and coordinating interactions between controllers and repositories.

6.3 Repository Layer

Repositories abstract database operations, allowing the backend to interact with the database without explicit SQL queries.

6.4 Persistence Layer (Entity Classes)

Entity classes define the data model and its mapping to the relational database, ensuring a consistent schema.

6.5 Applied Design Principles (DTOs)

Data Transfer Objects (DTOs) enhance encapsulation and optimize data transfer between layers and external clients.

7 Area Borders

The area borders feature addresses the research question by implementing computational geometry algorithms for precise geographical boundary calculations.

7.1 Purpose of Area Borders in the App

Accurate area borders are essential for defining regions based on user input, supporting the app's mapping functionality.

7.2 Overview of the Convex Hull Algorithm

The convex hull algorithm identifies the smallest convex polygon enclosing a set of points, making it a suitable choice for this project.

7.3 Use Cases of the Convex Hull in Industry

Applications of convex hulls in mapping, computer graphics, and robotics highlight their importance in solving real-world problems.

7.4 Alternate Methods for Area Border Calculation

Alternative methods like Voronoi diagrams and alpha shapes were considered but found less suitable due to complexity or computational demands.

7.5 Rationale for Choosing the Convex Hull Method

The convex hull algorithm offers a balance of simplicity, efficiency, and accuracy, aligning with the project's requirements.

7.6 Integration of the Algorithm into the Backend

The algorithm is implemented in the service layer, ensuring smooth integration with other backend components.

7.7 Challenges and Adjustments

Challenges included handling edge cases like collinear points, which were resolved through specific algorithm adjustments.

8 Implementation

The backend implementation combines theoretical concepts with practical solutions to ensure functionality and scalability.

8.1 Config of Spring Boot (application.properties)

The `application.properties` file configures essential settings, including database connections, logging, and server parameters.

8.2 Entity Classes (Structure/Purpose)

Entity classes define the application's data model, using annotations to map fields to database tables.

8.3 JpaRepositories (DB Access and CRUD Operations)

Repositories simplify database access by providing methods for CRUD operations and enabling custom queries.

8.4 Service Classes

Service classes encapsulate business logic, coordinating data flow between controllers and repositories.

8.5 Rest Controller (API Endpoints and their Functions)

REST controllers define API endpoints, processing requests and returning responses to ensure seamless interaction with the frontend.

9 Final Thoughts

9.1 Leon Edlinger

9.2 Paul Gigler

9.3 Andreas Weissl

10 Meetings

Protokolle der Meetings, vielleicht auch ein zeitplan wann immer und wie lang

11 Working Hours

Arbeitspaket-Nr.	Beschreibung	Dauer
1	Einführung und Einarbeitung	8 h
2	Grundkonzept erstellen	8 h
3	Struktur der App festlegen	6 h
5	Wifi-Socket in App implementieren	39 h
6	Write-Funktionalität in App implementieren	14 h
7	Read-Funktionalität in App implementieren	19 h
8	Trim-Funktionalität in App implementieren	10 h
9	Konfigurationsmöglichkeiten für Flug in App implementieren	16 h
10	Höhenregelung-Funktionalität in App implementieren	14 h
12	Graphische Darstellung der Flugdaten	18 h
14	App testen und debuggen	19 h
26	Gesamtkonzept testen und debuggen	16 h
Summe		**187 h**

Table 1: Arbeitszeitnachweis

12 Source code directory

Source Code directory, kein plan was des is

13 List of figures

1 Dart Logo (Source: https://dart.dev) 3
2 Flutter Logo (Source: https://flutter.dev/) 3
3 Java Logo (Source: https://logodownload.org/java-logo/) 4
4 Git Logo (Source: https://git-scm.com/) 5
5 GitHub Logo (Source: https://github.com/) 5
6 OSM Logo (Source: https://wiki.openstreetmap.org/) 6
7 GraphHopper Logo (Source: https://brandfetch.com/graphhopper.com) 6
8 VS Code Logo (Source: https://code.visualstudio.com/brand) 7
9 IntelliJ IDEA Logo (Source: https://www.jetbrains.com/company/brand/) 7
10 Android Studio Logo (Source: https://img.icons8.com/?size=100&id=04OFrkjznvcd&format=png&color=000000) 8
11 Postman Logo (Source: https://www.postman.com/legal/logo-usage/) 8
12 Docker Logo (Source: https://www.docker.com/) 9

14 List of tables

1 Arbeitszeitnachweis 15

15 Bibliography

[1] *About GitHub and Git - GitHub Docs*. [Online; accessed 16. Feb. 2025]. Feb. 2025. URL: https://docs.github.com/en/get-started/start-your-journey/about-github-and-git.

[2] Contributors to Wikimedia projects. *IntelliJ IDEA - Wikipedia*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://en.wikipedia.org/w/index.php?title=IntelliJ_IDEA&oldid=1272820016.

[3] Lukas Dagne. "Flutter for cross-platform App and SDK development". In: (2019).

[4] *Flutter for Beginners*. URL: https://books.google.at/books?hl=de&lr=&id=pF6vDwAAQBAJ&oi=fnd&pg=PP1&dq=benefits+dart+language&ots=dZJWUGVs4x&sig=a196WqhXmQzuy23cmcKpEpIqn_k&redir_esc=y#v=onepage&q=benefits%20dart%20language&f=false.

[5] *flutter/README.md at master · flutter/flutter*. [Online; accessed 23. Jan. 2025]. Jan. 2025. URL: https://github.com/flutter/flutter/blob/master/README.md.

[6] *Git - Branching and Merging*. [Online; accessed 16. Feb. 2025]. Feb. 2025. URL: https://git-scm.com/about/branching-and-merging.

[7] *graphhopper*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://github.com/graphhopper/graphhopper.

[8] *Host Your Own Worldwide Route Calculator With GraphHopper - GraphHopper Directions API*. [Online; accessed 30. Jan. 2025]. Apr. 2024. URL: https://www.graphhopper.com/blog/2022/06/27/host-your-own-worldwide-route-calculator-with-graphhopper.

[9] *IntelliJ IDEA – the Leading Java and Kotlin IDE*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://www.jetbrains.com/idea.

[10] *OSM: The simple map that became a global movement*. [Online; accessed 28. Jan. 2025]. Jan. 2025. URL: https://web.archive.org/web/20240420144212/https://www.directionsmag.com/article/1163.

[11] *Quickstart for GitHub Actions - GitHub Docs*. [Online; accessed 16. Feb. 2025]. Feb. 2025. URL: https://docs.github.com/en/actions/writing-workflows/quickstart.

[12] *Stack Overflow Developer Survey 2022*. [Online; accessed 16. Feb. 2025]. Feb. 2025. URL: https://survey.stackoverflow.co/2022/#version-control-version-control-system.

[13] *Visual Studio Code for the Web*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://code.visualstudio.com/docs/editor/vscode-web.

[14] *vscode*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://github.com/microsoft/vscode.

[15] *What is the Visual Studio Code editor built on*. [Online; accessed 30. Jan. 2025]. Jan. 2025. URL: https://stackoverflow.com/questions/29966093/what-is-the-visual-studio-code-editor-built-on.

[16] OpenStreetMap Wiki. *Main Page — OpenStreetMap Wiki*. [Online; accessed 28-January-2025]. 2024. URL: https://wiki.openstreetmap.org/w/index.php?title=Main_Page&oldid=2752444.

16 Abbreviation

VCS	Version Control System
API	Application Programming Interface
CI|CD	Continuous Integration & Continuous Deployment
OSM	Open Street Map
GIS	Geo Information System
GH	GraphHopper
IDE	Integrated Development Environment
GPS	Global Positioning System
GUI	Graphical User Interface
ADB	Android Debugging Bridge
USB	Universal Serial Bus
REST	REpresentational State Transfer
JVM	Java Virtual Machine