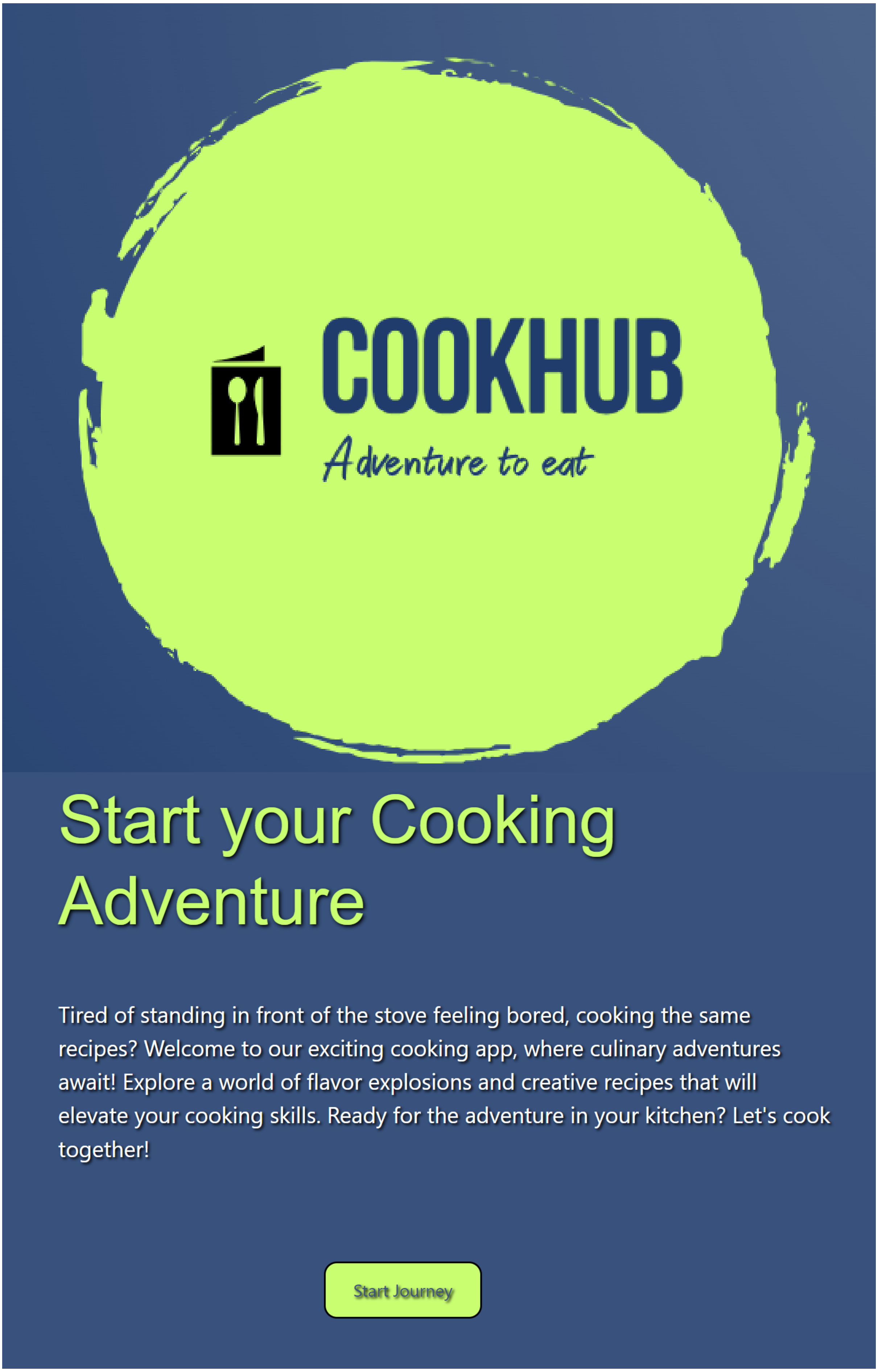


COOKHUB

Adventure to eat

Start your Cooking Adventure

Tired of standing in front of the stove feeling bored, cooking the same recipes? Welcome to our exciting cooking app, where culinary adventures await! Explore a world of flavor explosions and creative recipes that will elevate your cooking skills. Ready for the adventure in your kitchen? Let's cook together!

CookHub

Product Vision

CookHub is intended to be an **online recipe book** that can **store its own recipes** in a database, **suggest recipes** via the OpenAI API, can save your **own created recipes**, and **create a shopping list** from your favorite cooking algorithm. In addition, the OpenAI API allows a tasty recipe to be turned into an **EATventure** Ⓒ by wrapping the cooking instructions in an **automatically generated adventure story** to turn simple cooking into a real experience.

What we plan to deliver

A functional web app which can replace all heavy cooking books, handwritten dusty recipes and hopefully all other cooking apps of our customers in the long term. Because we've got more than just recipes - **we've got EATventures** Ⓒ.

COOKHUB Adventure to eat - TEAM

Marc Siegfarth – Product Owner, Lead Developer

Nick Hörner – Creative Director, Senior Developer

Stefan Mergl - Scrum Master, Developer

Len Vejsada – Time-Tracking Manager, Developer

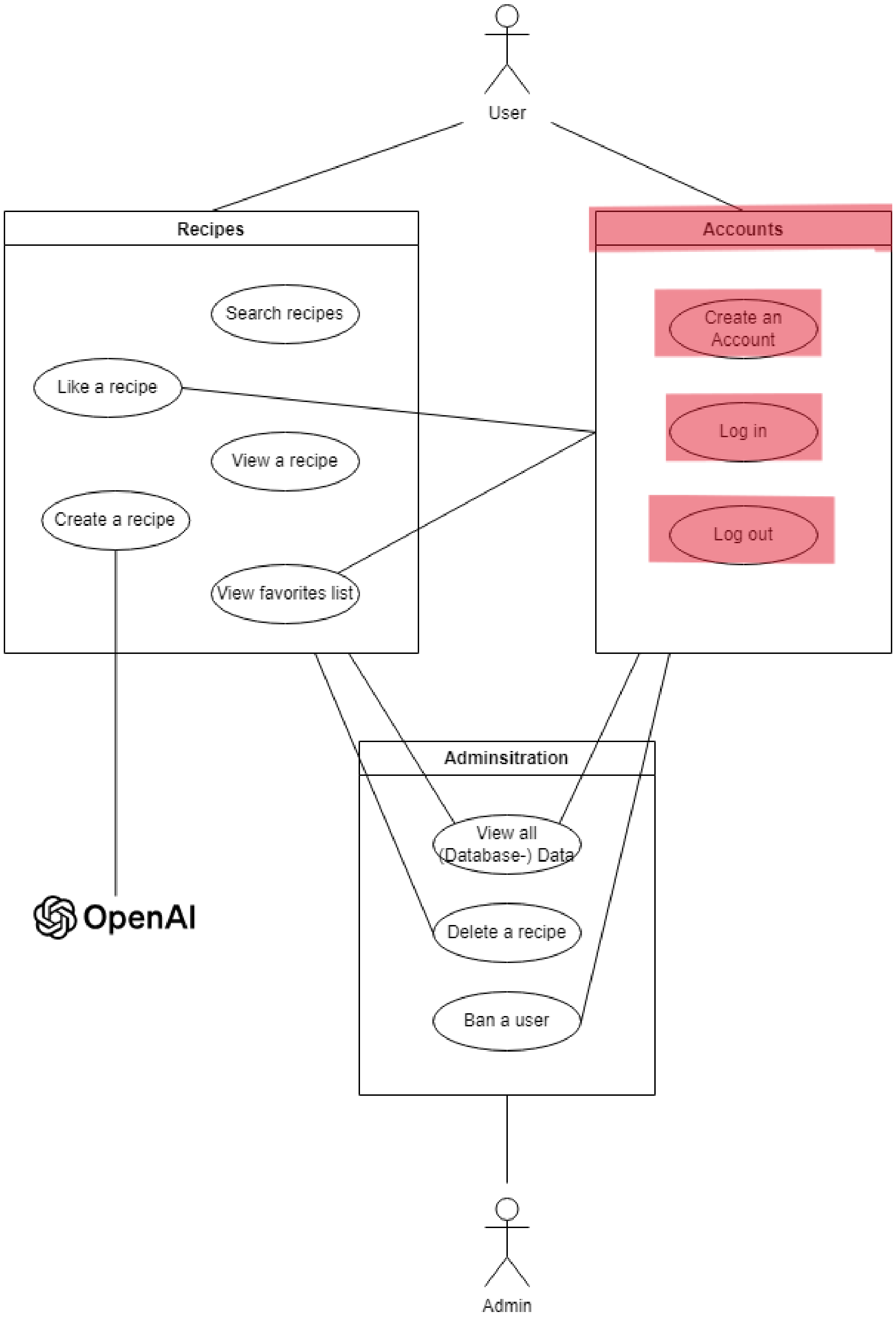
And here's our newest team member:

Chef Carlos!

He's the chef in the kitchen and guides and advices new community members. Furthermore he has become the face of **COOKHUB!**

Caaaarloooos!

Use case diagram:

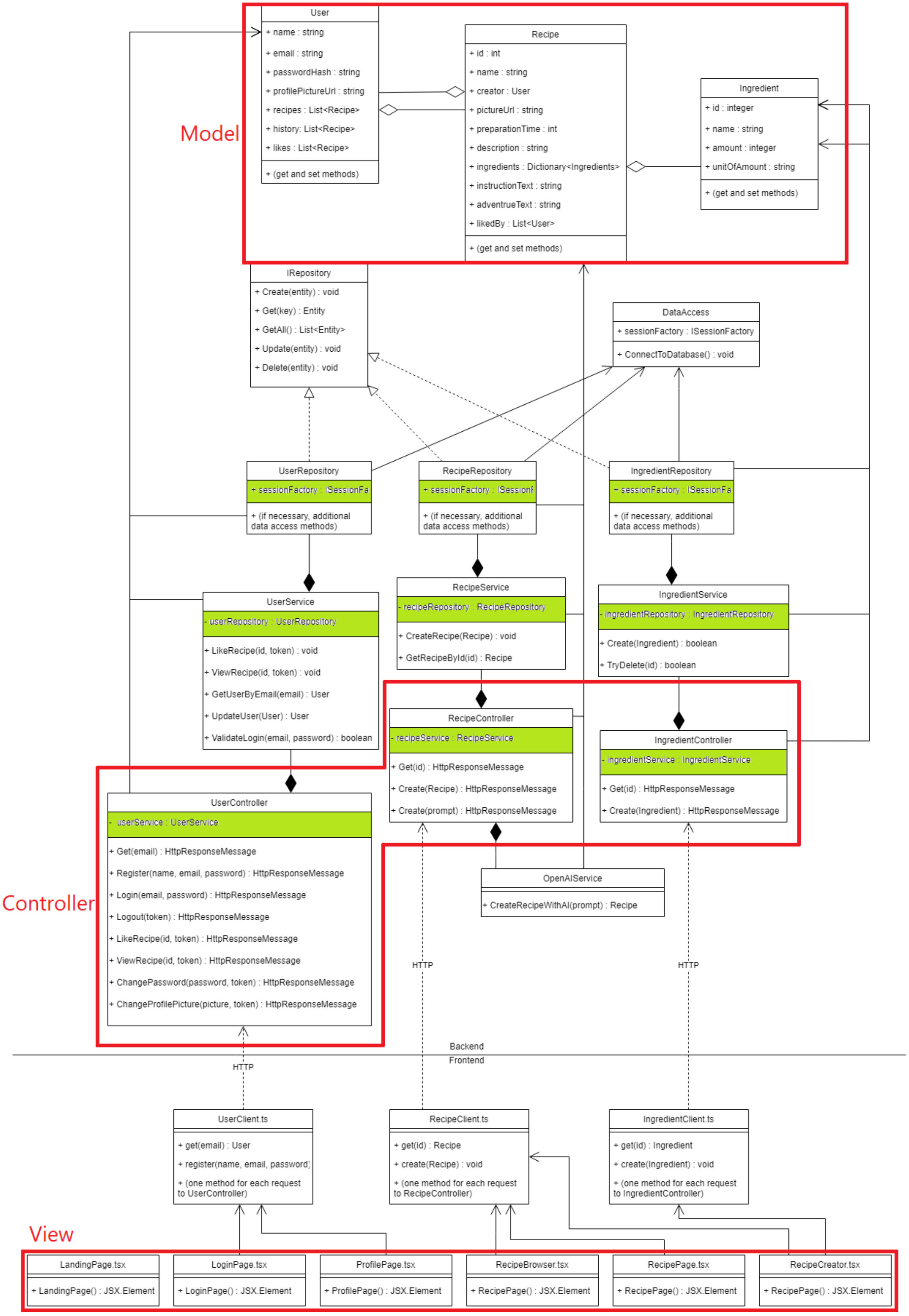


Focus so far:

Frontend and login functionality:

- **Create Account**
- **Log in**
- **Log out**

Class diagram & design principles:


Model
User
+ name : string
+ email : string
+ passwordHash : string
+ profilePictureUrl : string
+ recipes : List<Recipe>
+ history: List<Recipe>
+ likes : List<Recipe>
+ (get and set methods)
Recipe
+ id : int
+ name : string
+ creator : User
+ pictureUrl : string
+ preparationTime : int
+ description : string
+ ingredients : Dictionary<Ingredients>
+ instructionText : string
+ adventrueText : string
+ likedBy : List<User>
+ (get and set methods)
Ingredient
+ id : integer
+ name : string
+ amount : integer
+ unitOfAmount : string
+ (get and set methods)
IRepository
+ Create(entity) : void
+ Get(key) : Entity
+ GetAll() : List<Entity>
+ Update(entity) : void
+ Delete(entity) : void
DataAccess
+ sessionFactory : ISessionFactory
+ ConnectToDatabase() : void
UserRepository
+ sessionFactory : ISessionFa
+ (if necessary, additional data access methods)
RecipeRepository
+ sessionFactory : ISessionF
+ (if necessary, additional data access methods)
IngredientRepository
+ sessionFactory : ISessionFa
+ (if necessary, additional data access methods)
UserService
- userRepository : UserRepository
+ LikeRecipe(id, token) : void
+ ViewRecipe(id, token) : void
+ GetUserByEmail(email) : User
+ UpdateUser(User) : User
+ ValidateLogin(email, password) : boolean
RecipeService
- recipeRepository : RecipeRepository
+ CreateRecipe(Recipe) : void
+ GetRecipeById(id) : Recipe
IngredientService
- ingredientRepository : IngredientRepository
+ Create(Ingredient) : boolean
+ TryDelete(id) : boolean
Controller
UserController
- userService : UserService
+ Get(email) : HttpResponseMessage
+ Register(name, email, password) : HttpResponseMessage
+ Login(email, password) : HttpResponseMessage
+ Logout(token) : HttpResponseMessage
+ LikeRecipe(id, token) : HttpResponseMessage
+ ViewRecipe(id, token) : HttpResponseMessage
+ ChangePassword(password, token) : HttpResponseMessage
+ ChangeProfilePicture(picture, token) : HttpResponseMessage
RecipeController
- recipeService : RecipeService
+ Get(id) : HttpResponseMessage
+ Create(Recipe) : HttpResponseMessage
+ Create(prompt) : HttpResponseMessage
IngredientController
- ingredientService : IngredientService
+ Get(id) : HttpResponseMessage
+ Create(Ingredient) : HttpResponseMessage
OpenAIService
+ CreateRecipeWithAI(prompt) : Recipe
HTTP
HTTP
HTTP
Backend
Frontend
UserClient.ts
+ get(email) : User
+ register(name, email, password)
+ (one method for each request to UserController)
RecipeClient.ts
+ get(id) : Recipe
+ create(Recipe) : void
+ (one method for each request to RecipeController)
IngredientClient.ts
+ get(id) : Ingredient
+ create(Ingredient) : void
+ (one method for each request to IngredientController)
View
LandingPage.tsx
+ LandingPage() : JSX.Element
LoginPage.tsx
+ LoginPage() : JSX.Element
ProfilePage.tsx
+ ProfilePage() : JSX.Element
RecipeBrowser.tsx
+ RecipePage() : JSX.Element
RecipePage.tsx
+ RecipePage() : JSX.Element
RecipeCreator.tsx
+ RecipePage() : JSX.Element

Models: Classes that represent our database entities
Views: React components that define our different web pages
Controllers: Backend API endpoints, which take requests by the frontend and change/transfer the requested data

To achieve dependency inversion, we use the built-in IoC container of ASP.NET. This will automatically inject dependencies through the constructor of our classes (after a little bit of configuration).

Software tools/platforms/techniques:

CI/CD:

- *Git* (GitHub)
- *GitHub actions*

Project management tools:

- *TMetric* for time tracking
- *draw.io* for diagrams
- *Figma* for mockup designs

IDEs:

- *WebStorm* for Frontend/TypeScript
- *Rider* for Backend/C#
- IntelliJ for everything

Frontend technologies:

- language: *TypeScript*
- libraries: *React*, *Material UI*
- *Locofy (Figma AI Converter)*

Backend technologies:

- language: *C#*
- Frameworks & Libraries: *ASP.NET Core* (for API calls & dependency injection), *NHibernate* (for database/ORM-mapping), *xUnit.NET* (for testing), *SQLite* (for test database)

Database:

- *PostgreSQL*

Additional Technologies:

- ChatGPT4 (adventurize it prompt testing)
- OpenAI API

Work Techniques:

- Agile Working (SCRUM)
- Weeklys
- Issue Tracking
- Usage of Kanban Board
- Peer Review before Merge in main branch

COOKHUB
Adventure to eat

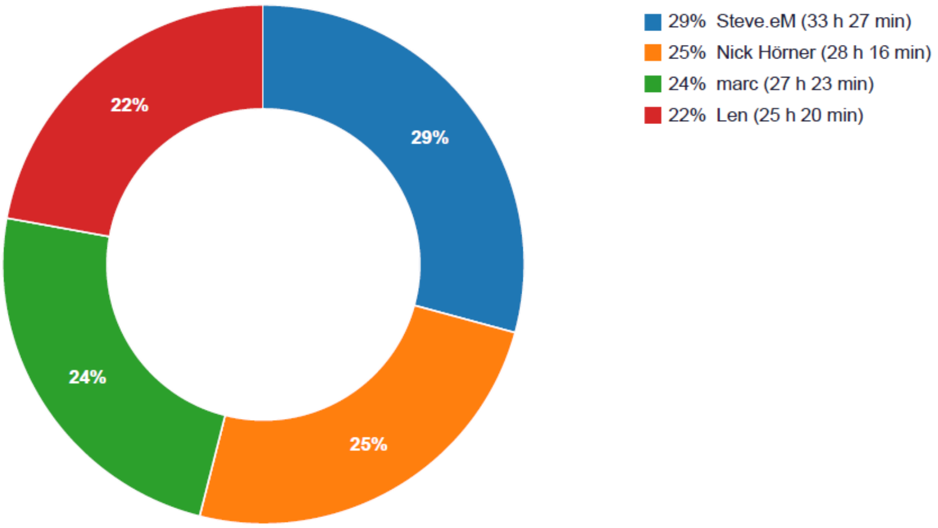
TMETRIC

Aufgaben-Bericht

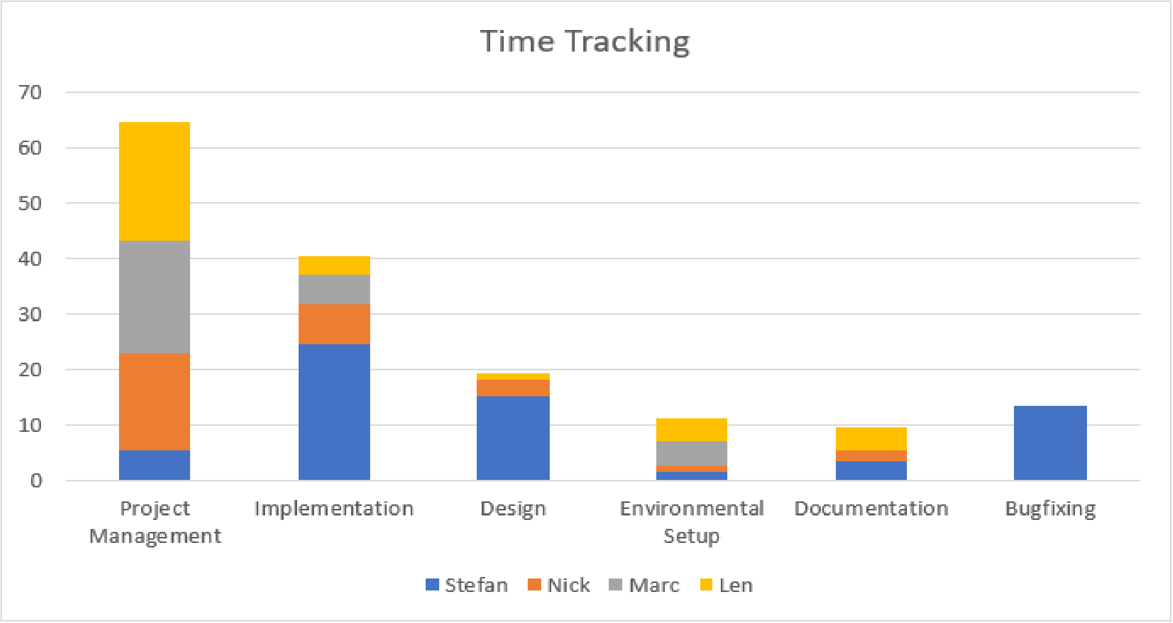
05.10.2023 - 30.11.2023

Gesamtzeit 114 h 26 min

Gesamtzeit Verteilung für Benutzer



Workflow	**Time**
Project Management	61,15h
Implementation	29,59h
Design	13,19h
Environmental Setup	11,49h
Documentation	7,06h
Bugfixing	1,50h


Time Tracking
Project Management
Implementation
Design
Environmental Setup
Documentation
Bugfixing
Stefan
Nick
Marc
Len