

**Smart Study Timer**

**Документация**

## Съдържание

1	Въведение.....	3
1.1	Цел.....	3
1.2	Резюме.....	3
1.3	Дефиниции и акроними .....	3
2	Използвани технологии .....	4
3	Реализация на базата от данни.....	5
4	Реализация на бизнес логиката .....	7
5	Реализация на потребителския интерфейс.....	11
6	Внедряване на системата .....	18

# 1 ВЪВЕДЕНИЕ

## 1.1 Цел

Целта на този документ е да опише в детайли реализацията на системата **Smart Study Timer** – приложение, което предоставя интелигентни инструменти за подпомагане на ефективното управление на учебното време. Системата **Smart Study Timer** е разработена с цел да подпомогне ефективното управление на времето за учене, като използва технологични решения за организация и подобряване на концентрацията. Тя е насочена към ученици и/или студенти, които се стремят към по-добри резултати в своята учебна подготовка.

## 1.2 Резюме

Основните функционалности на системата **Smart Study Timer** включват таймер за учене, както и вградена поддръжка за възпроизвеждане на релаксираща музика, включително звуци като бял шум (white noise) и кафяв шум (brown noise), с цел създаване на подходяща учебна атмосфера. Системата **Smart Study Timer** е проектирана с гъвкава архитектура, която да отговаря на индивидуалните нужди на потребителите и да позволява лесно мащабиране при растящо натоварване. Тя е съставена от множество модули, които взаимодействат помежду си чрез принципите на многослойните архитектури и REST API интерфейси, а именно:

### Timer Management Module

- Реализира основната функционалност на приложението – управление на учебни таймери.
- Включва REST API за създаване, стартиране, спиране и анализиране на учебни сесии.
- Дава възможност за запис и извличане на данни за минали сесии.

### User Management Module

- Управлява идентификацията на потребителите (регистрация и вход).
- Интегриран е със Spring Security.
- Отговаря за създаването и управлението на потребителски профили.
- Включва логика за валидация на входящи данни.


### UI Module


- Представява frontend частта, реализирана с помощта на HTML, CSS и JavaScript.
- Интеграция с backend чрез HTTP заявки.
- Предоставя функционалност за логване, регистрация, създаване на учебни сесии, визуализация на статистики и управление на профил.

### Database Layer

- Използва MySQL за съхранение на данните.
- Структуриран в няколко таблици (потребители, таймери за изминалия ден, таймери за изминалата седмица).
- Интегриран с Hibernate за автоматизирано управление на базата от данни.

## 1.3 Дефиниции и акроними

 **REST API** – Интерфейс за програмиране на приложения, използващ HTTP протокола за взаимодействие между клиент и сървър.

 **База от данни** – Организирана колекция от данни, която улеснява управлението и извличането на информация.

- ✚ **Потребителски интерфейс (UI)** – Графичен интерфейс, който позволява взаимодействие между потребителя и системата.
- ✚ **Бизнес логика** – Логическа обработка на данните, която прилага правилата и функционалностите на приложението.
- ✚ **Сървър** – Софтуер, който приема заявки от клиенти и връща отговори.
- ✚ **Клиент** – Софтуерна или хардуерна система, която прави заявки към сървър.
- ✚ **Бял шум** – Сигнал или звук, който съдържа всички честоти в звуковия спектър с еднаква енергия на честота. Той е хаотичен и равномерно разпределен, наподобяващ статичния звук от радио или телевизор. Подходящ за среда с много разсейващи звуци, като офиси, общежития или шумни кафенета.
- ✚ **Кафяв шум** – Сигнал, при който енергията намалява с увеличаване на честотата, така че по-ниските честоти доминират. Той е по-дълбок и плавен, наподобяващ звуци като водопад или далечно море. По-добър за поддържане на съсредоточеност, особено при подготовка за изпити или четене в тиха среда.

## 2 ИЗПОЛЗВАНИ ТЕХНОЛОГИИ

- ✚ **Java 23** – Основен език за реализация на бизнес логиката.
- ✚ **Spring Boot** – Фреймуърк за разработка на уеб приложения и управление на зависимостите.
- ✚ **MySQL** – Релационна база от данни, използвана за съхранение на потребителски данни и времеви записи.
- ✚ **HTML, CSS, JavaScript** – Инструменти за създаване на графичния потребителски интерфейс.
- ✚ **Maven** – Инструмент за управление на зависимостите и изграждане на проекта.

### 3 РЕАЛИЗАЦИЯ НА БАЗАТА ОТ ДАННИ

В системата **Smart Study Timer** се използва реляционна база от данни **MySQL**, която съхранява данните за потребителите, техните таймери и свързаната с тях информация.

Базата от данни **userDatabase** има следната структура:

#### 3.1. Таблици

##### 3.1.1. Таблица: **users**

Тази таблица съдържа информация за регистрираните потребители.

Поleta:

- **id** – Уникален идентификатор на потребителя (Основен ключ).
- **email** – Имейл адрес на потребителя.
- **first\_name** – Име на потребителя.
- **last\_name** – Фамилия на потребителя.
- **password** – Хешираната парола на потребителя.
- **username** – Потребителско име на потребителя.

##### 3.1.4. Таблица: **current\_day\_timers**

Тази таблица съхранява данни за таймерите, проследяващи учебната активност за конкретен ден.

Поleta:

- **id** – Уникален идентификатор на таймера (Основен ключ).
- **current\_day** – Дата, за която се записват данни от таймера (например "2025-01-21").
- **number\_of\_study\_sessions** – Броят на учебните сесии, проведени през този ден.
- **time\_spent\_studying** – Общо време, прекарано в учене през този ден (в минути или секунди).
- **username** – Потребителското име на потребителя, за когото се отнасят данните. Това е външен ключ, сочещ към таблицата **users**.

##### 3.1.5. Таблица: **current\_week\_timers**

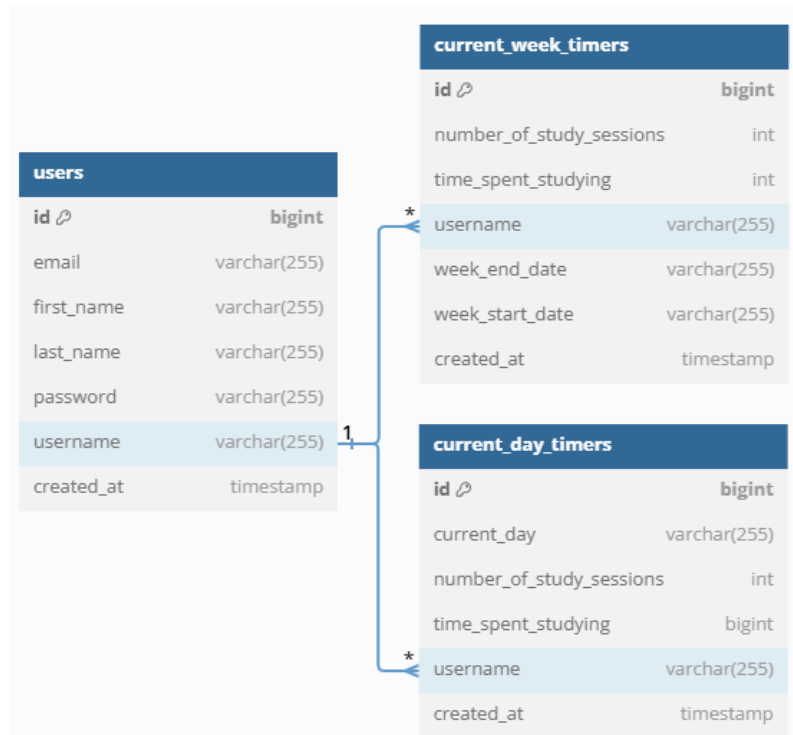
Тази таблица съдържа данни за таймерите, проследяващи учебната активност през текущата седмица.

Поleta:

- **id** – Уникален идентификатор на таймера (Основен ключ).
- **number\_of\_study\_sessions** – Броят на учебните сесии, проведени през седмицата.
- **time\_spent\_studying** – Общо време, прекарано в учене през седмицата.
- **username** – Потребителското име на потребителя, за когото се отнасят данните. Това е външен ключ, сочещ към таблицата **users**.

- **week\_start\_date** – Начална дата на седмицата (например "2025-01-01").
- **week\_end\_date** – Краен срок на седмицата (например "2025-01-07").

Диаграма на цялата база от данни, където можем да видим прегледно релациите между различните таблици:



## 4. РЕАЛИЗАЦИЯ НА БИЗНЕС ЛОГИКАТА

Бизнес логиката на системата Smart Study Timer е реализирана с помощта на Java Spring Framework и Spring Boot. Системата е разделена на два основни модула, а именно **UserManager** и **TimerManager**, които управляват съответно потребителите и времето за учене.

### 1. Модул UserManager

Модулът UserManager е отговорен за управлението на потребителите в системата и включва функционалности за регистрация, логин и управление на профил. Структурата на този модул включва:

#### 1. Configurations Layer:

- **JpaConfig** - Конфигурация за настройка на JPA (Java Persistence API), която управлява връзката с базата данни и дефинира основни настройки за работа с ORM.
- **SecurityConfig** - Конфигурация за сигурността на приложението, която настройва правилата за автентикация и авторизация.
- **WebConfig** - Конфигурация за настройка на уеб средата на приложението.

#### 2. Controllers Layer:

- **UserController** – Отговаря за регистрацията и логина на потребителите.
- **LogoutController** – Обработва процеса на логване извън системата.
- **MainController** – Зарежда главната страница на приложението.
- **ProfileController** – Управлява операциите за изобразяване и редактиране на профила на потребителя.
- **RegistrationController** – Обработва заявките за нови потребители.

#### 3. Service Layer:

- **UserService** – Съдържа бизнес логиката за управление на потребителските акаунти, включително валидиране на данни при регистрация и логин.

#### 4. Repository Layer:

- **UserRepository** – Репозитори за работа с базата данни, което осигурява CRUD операции за таблицата с потребители.
- **UserAssembler** - Преобразува данни между Model и DTO, като улеснява манипулацията и трансфера на данни между различните слоеве на приложението.

#### 5. DTO Layer:

- **UserDTO, LoginDTO, ProfileDTO, UserProfileDTO** – Класове, които се използват за предаване на данни между различните слоеве на приложението.

#### 6. Model Layer:

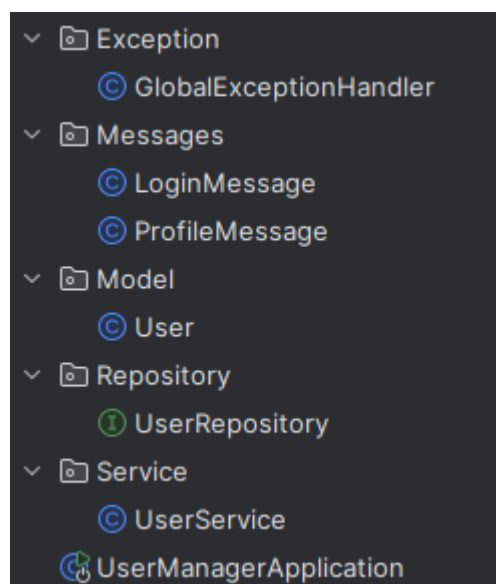
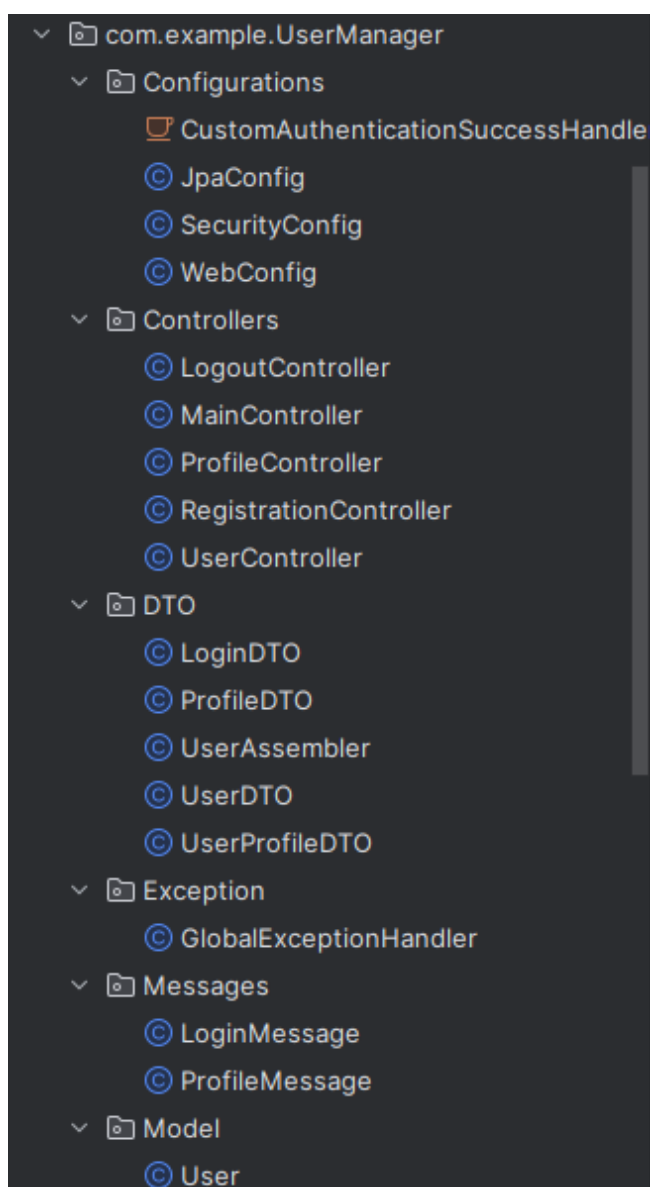
- **User** - представя данни за потребителите, като включва полета като id, email, first\_name, last\_name, password, username и created\_at, и се използва за взаимодействие с базата данни и други слоеве на приложението.

## 7. Exception Handling:

- **GlobalExceptionHandler** – Глобален обработчик на изключения, който улавя грешки по време на работата на системата, например при невалидни входни данни.

## 8. Messages Layer:

- **LoginMessage** - Съобщение, което съдържа информация, свързана с процеса на вход в приложението, като например състоянието на автентикацията или грешки при вход.
- **ProfileMessage** - Съобщение, което съдържа данни за потребителския профил, като информация за потребителя.





## 2. Модул TimerManager

Модулът TimerManager отговаря за управлението на таймерите и измерването на времето за учене. Структурата му включва следните слоеве:

### 1. Configurations Layer:

- **JpaConfig** - Конфигурация за настройка на JPA (Java Persistence API), която управлява връзката с базата данни и дефинира основни настройки за работа с ORM.
- **SecurityConfig** - Конфигурация за сигурността на приложението, която настройва правилата за автентикация и авторизация.
- **WebConfig** - Конфигурация за настройка на уеб средата на приложението.

### 2. Controller Layer:

- **CurrentDayTimerController** – Управлява учебните сесии за текущия ден.
- **CurrentWeekTimerController** – Управлява времето за учене за цялата седмица.
- **WeekCalculator** – Изчислява общото време за учене през седмицата.

### 3. Service Layer:

- **CurrentDayTimerService** и **CurrentWeekTimerService** – Съдържат логиката за актуализиране на времето за учене на потребителя в реално време.

### 4. Repository Layer:

- **CurrentDayTimerRepository** и **CurrentWeekTimerRepository** – Репозитори за работа с данни за таймерите в базата данни, които извършват CRUD операции за съхранение и извличане на сесиите за учене.
- **CurrentDayTimerAssembler** и **CurrentWeekTimerAssembler**

### 5. DTO Layer:

- **CurrentDayTimerDTO** и **CurrentWeekTimerDTO** – Съдържат данни за сесиите на текущия ден и седмицата за учене.

### 6. Model Layer:

- **CurrentDayTimer** и **CurrentWeekTimer** – Модели, които представят данни за времето, прекарано в учене на дневна и седмична база.

### 7. Message Layer:

- **ProfileMessage** и **ProfileMessage2** – Дават информация за успешното завършване на операцията за измерване на времето съответно за деня и за седмицата.

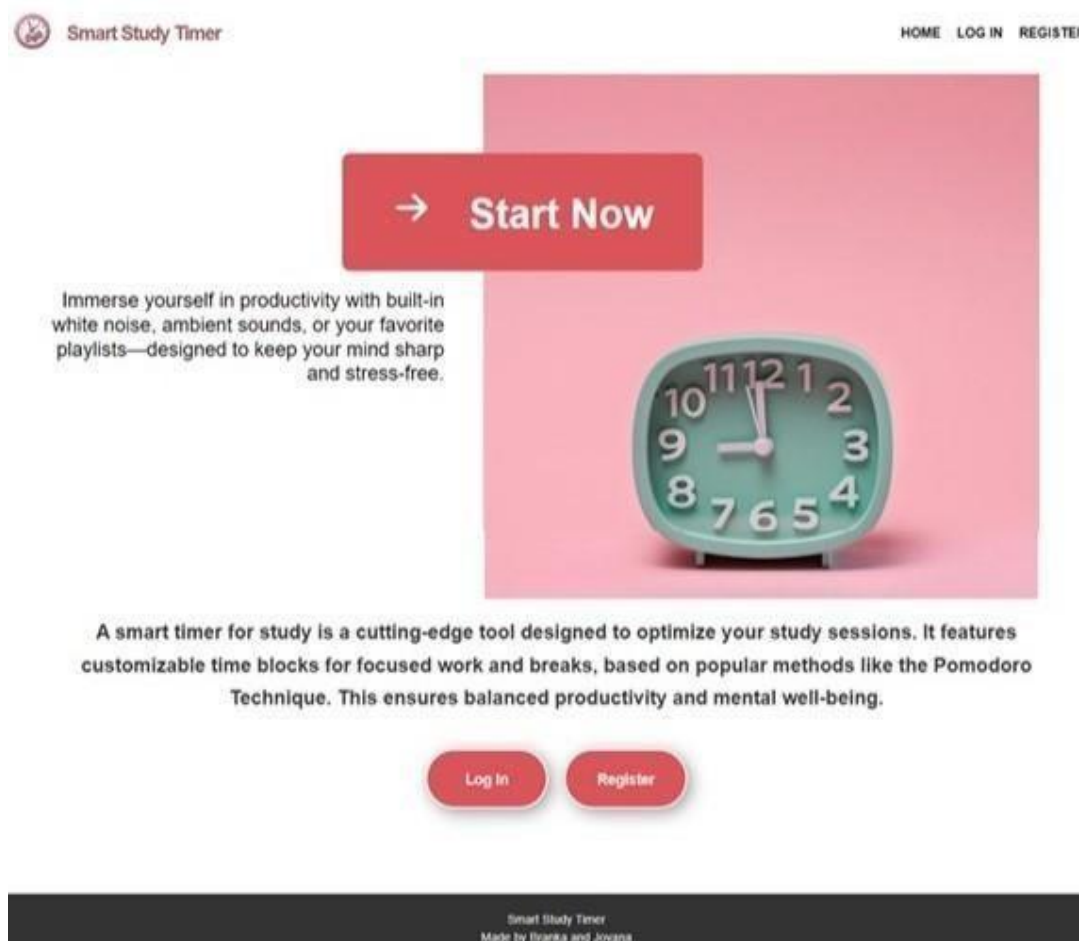
- com.example.TimerManager
  - Configuration
    - JpaConfig
    - SecurityConfig
    - WebConfig
  - Controllers
    - CurrentDayTimerController
    - CurrentWeekTimerController
    - ProfileController
    - TimerController
    - WeekCalculator
  - DTO
    - CurrentDayTimerAssembler
    - CurrentDayTimerDTO
    - CurrentWeekTimerAssembler
    - CurrentWeekTimerDTO
    - ProfileDTO
    - TimerAssembler
    - TimerDTO
  - Message
    - ProfileMessage
    - ProfileMessage2

- Model
  - CurrentDayTimer
  - CurrentWeekTimer
  - Timer
- Repository
  - CurrentDayTimerRepository
  - CurrentWeekTimerRepository
  - TimerRepository
- Service
  - CurrentDayTimerService
  - CurrentWeekTimerService
  - TimerService
- TimerManagerApplication

## 5. РЕАЛИЗАЦИЯ НА ПОТРЕБИТЕЛСКИЯ ИНТЕРФЕЙС

### 1. Начална страница

Първото нещо, което потребителят вижда при стартиране на приложението, е началната страница. Тя изглежда така:



- След като е влязъл в началната страница, потребителят има две основни опции:
  - **Регистрация:** Ако потребителят няма профил в системата, най-напред е необходимо да се регистрира. До страницата с регистрационната форма може да стигне като натисне полето REGISTER в горния десен ъгъл, или като натисне бутона Register, намиращ се най-долу на страницата. Същото може да се постигне и като натисне бутона Start Now, който ще го пренасочи към страницата за вход, от където пак ще може да се регистрира натискайки върху полето “Don’t have a profile yet? Go ahead and create one now!”:

## Register

First Name:

Last Name:

Username:

Email:

Password:

Submit

You already have a profile?

- **Вход:** Ако потребителят вече има профил, може да натисне върху хиперлинка “You already have a profile?”, намиращ се най-долу в регистрационната форма.

## Log in

Username:

Password:

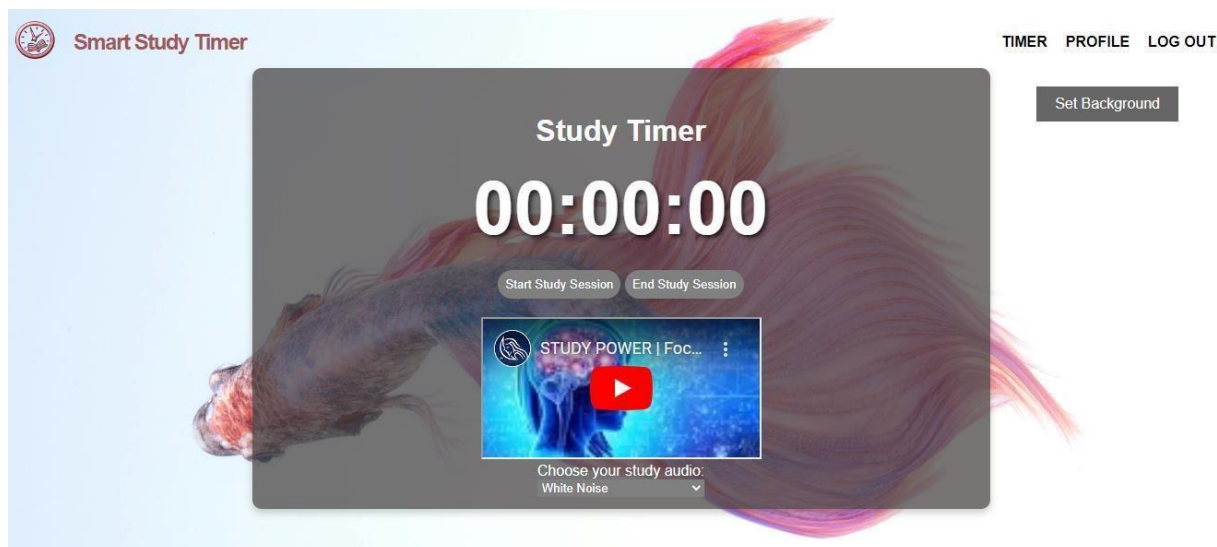
Submit

Don't have a profile yet? Go ahead and create one now!

Той ще го препрати към страницата за вход.

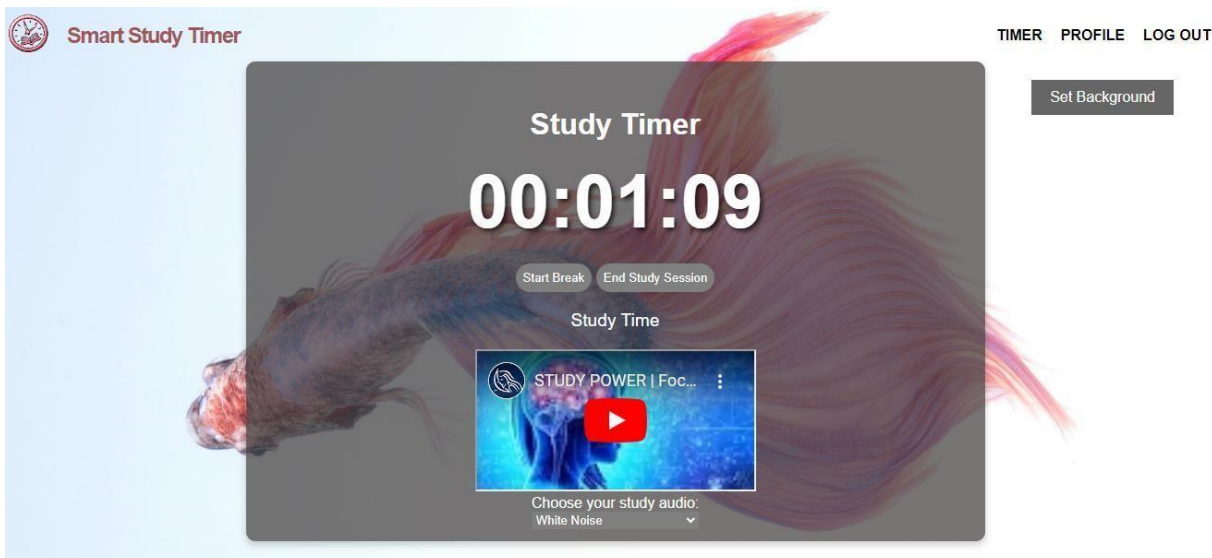
## 2. Страница за стартиране на учебната сесия

След като потребителят се логне успешно, се отваря страницата за стартиране на сесията за учене:



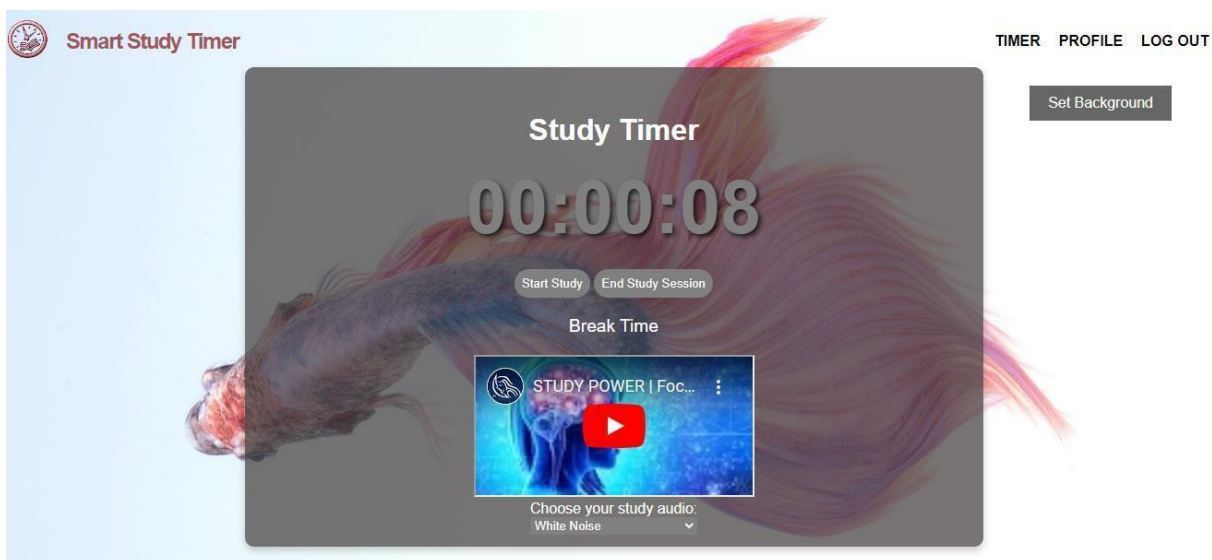
## 3. Стартиране на учебна сесия

За да започне учебната сесия, потребителят трябва да натисне бутона „Study“. Тогава се появява съобщението „Study Time“, което показва, че времето за учене е стартирало. Подобно, таймерът започва да отброява времето за учене.



#### 4. Почивка

Когато потребителят реши да си вземе почивка, трябва да натисне върху бутона „**Break**“. След натискане на този бутон, таймерът започва да отчита времето, прекарано в почивка, като сменя цвета си от бял в светлосив. Допълнително, появява се съобщението „Break Time“, което показва, че времето за почивка е стартирало.

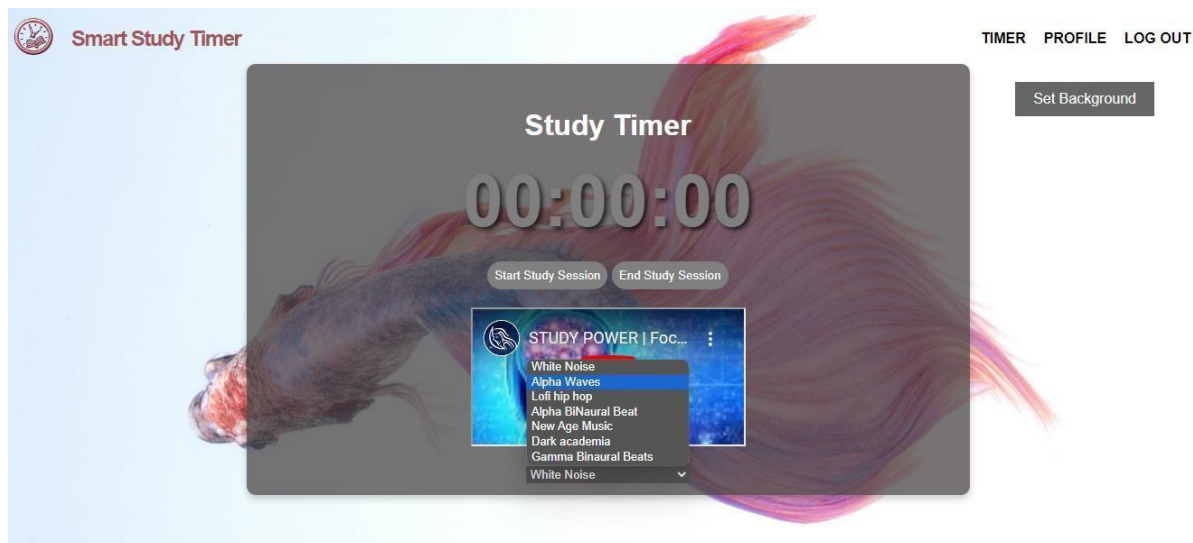


#### 5. Край на учебната сесия

Ако потребителят реши, че е постигнал достатъчно за деня, може да приключи учебната сесия, като натисне върху бутона „**End Study Session**“. След това таймерът се актуализира и показва общата продължителност на учебната сесия, взимайки предвид както времето за учене, така и времето за почивки. „Break Time“ и „Study Time“ вече няма да са видими на екрана до момента на стартиране на следващата учебна сесия.

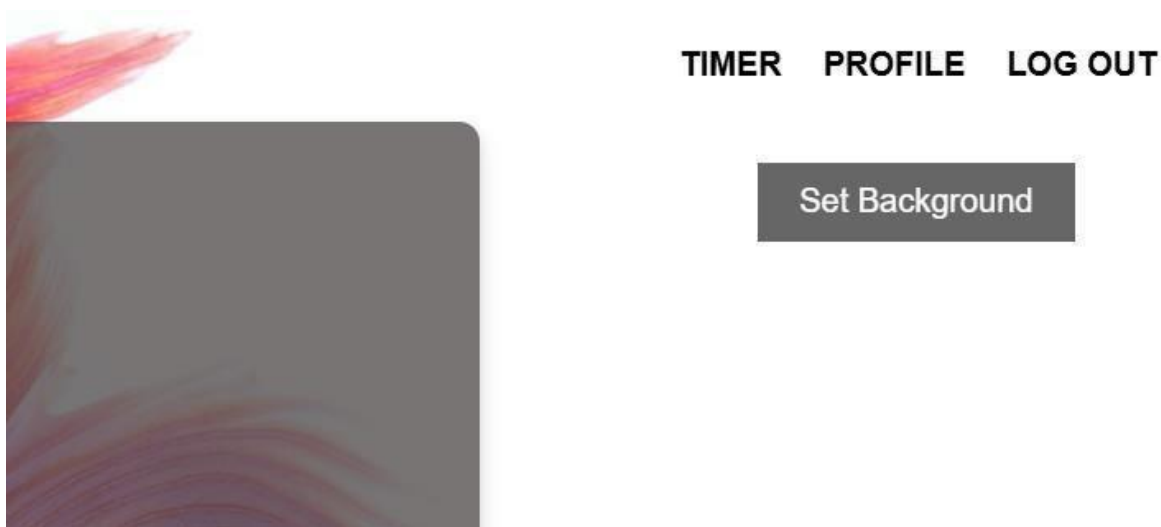
#### 6. Видео записи и музика

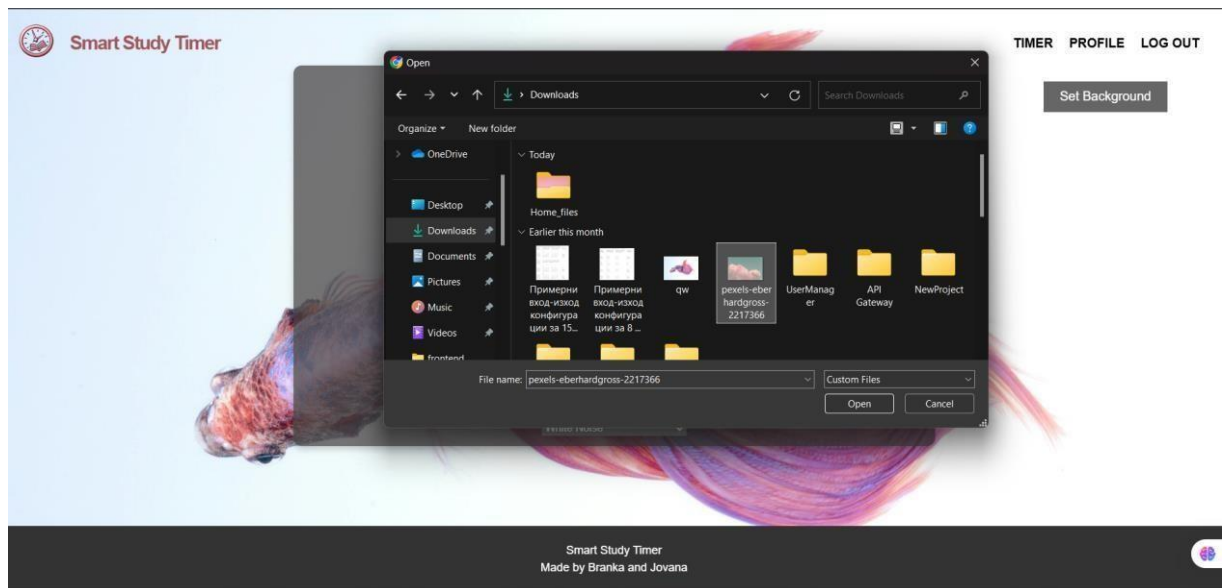
Приложението предоставя възможност на потребителя за избор на аудио, включително видео запис (бял шум, кафяв шум, или друго по негов избор), който да насърчи интензивна концентрация и спокойствие по време на учебната сесия. Това може да се направи, като се избере една от аудио опциите налични в падащото меню, намиращо се под „**Choose your study audio:**“.



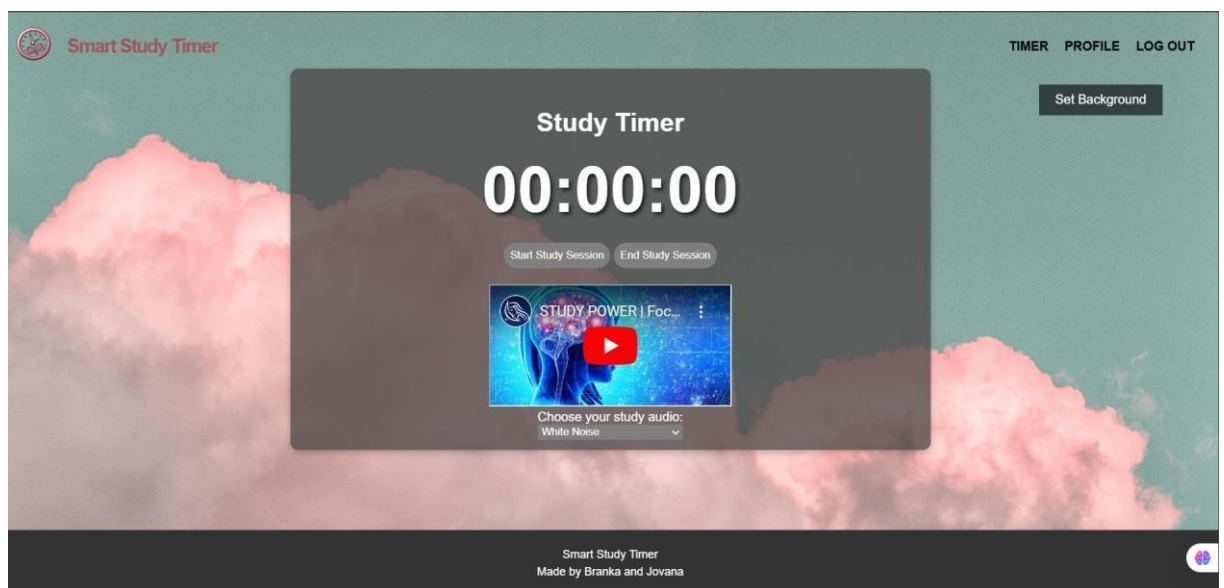
#### 7. Персонализиране на външния вид

Потребителят има опция за добавяне на нова фонова снимка (background). Това се осъществява с един клик върху бутона „**Set Background**“, след което потребителят може да избере снимка от своето устройство и така да направи учебната атмосфера още по-индивидуализирана спрямо своите нужди.





След избора на нов фон промените се отразяват незабавно и интерфейсьт изглежда така:



## 8. Профил на потребителя

Чрез натискане на полето за профил (PROFILE) в горния десен ъгъл, потребителят може да отвори своя личен профил. В профила се съхранява както личната информация за потребителя (име, фамилия, потребителско име, имейл адрес), така и времето, което е прекарал в учене за деня или седмицата.



## User Information

### User

**First Name:** TestUser

**Last Name:** Test

**Email:** test123@gmail.com

**Username:** TestUser

### Studying for the day

**Current day:** 21/01/2025

**Time spent studying:** 0h 0m 5s

**Number of study sessions:** 1

### Studying for the week

**Current week:** 2025-01-19 2025-01-25

**Time spent studying:** 0h 0m 5s

**Number of study sessions:** 1

## 6. ВНЕДРЯВАНЕ НА СИСТЕМАТА

### 1. Инсталиране на необходимите зависимости чрез Maven

- 1.1. Отворете проекта в IDE (например IntelliJ IDEA или Eclipse).
- 1.2. Уверете се, че **pom.xml** файлът съдържа всички необходими зависимости за проекта, които включват Spring Boot, Spring Data JPA, бази данни и други необходими библиотеки.
- 1.3. Изпълнете командата **mvn clean install** в терминала, за да изтеглите и инсталирате всички необходими библиотеки, посочени в **pom.xml** файла.

### 2. Конфигуриране на базата от данни в application.properties

- 2.1. Отидете в директорията **src/main/resources** и отворете файла **application.properties**.
- 2.2. Добавете настройки за връзка с базата данни (MySQL).
- 2.3. Въведете вашите данни за базата данни (като потребителско име, парола и URL на базата данни), както и настройки за JPA, за да се извършва автоматично създаване на таблици в базата данни.
- 2.4. Запишете промените в **application.properties** файла.

### 3. Стартиране на приложението чрез NewProjectApplication.java

- 3.1. Отидете в директорията **src/main/java/com/example/UserManager** и отворете файла **UserManagerApplication.java**, направете същото и за **src/main/java/com/example/TimerManager** и отворете файла **TimerManagerApplication.java**.
- 3.2. Уверете се, че файлът съдържа основната точка за стартиране на приложението с анотацията **@SpringBootApplication**.
- 3.3. Стартирайте приложението чрез IDE (натиснете **Run**).
- 3.4. Ако всичко е настроено правилно, приложението ще се стартира и ще бъде достъпно на адрес <http://localhost:8081> за **UserManagerApplication.java** и на <http://localhost:8082> за **TimerManagerApplication.java**.
- 3.5. За стартиране на изгледите на проекта е нужно да бъде инсталиран Python. Стартираме python сървър като отворим Command Prompt, насочваме терминала с командата **cd** към директорията в която се намират изгледите на страниците, след което извършваме командата **python -m http.server 3000**. При стартиране на командата изгледите стават достъпни и могат да се достъпват на следния адрес <http://127.0.0.1:3000>.