## **TЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ**

**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

**по професия код 481020 „Системен програмист“**

**специалност код 4810201**  **„Системно програмиране“**

Тема: Социално web приложение – Дневник на пътешественика

Дипломант: Дипломен ръководител:

*Ирина Фархад Ахамад инж. Станка Далекова*

СОФИЯ

2 0 2 3



УВОД

Споделянето на преживявания се превръща в популярна тенденция през последните години, благодарение на напредъка в технологиите, който улеснява документирането и публикуването на тези преживявания онлайн. Социалните медийни платформи играят значителна роля в тази дейност, позволявайки на потребителите да споделят снимки, видеоклипове и истории с широка аудитория от приятели, познати и непознати.

С течение на времето и с развитието си, тези платформи придобиват различни аудитории и цели. Някои са насочени към конкретен тип съдържание, като снимки или кратки видеоклипове, докато други са по-гъвкави и позволяват по-широк набор от информация. Платформите могат да бъдат предназначени за актуализации в реално време и взаимодействия на живо или да се фокусират повече върху разказване на истории. Съществуват социални медии, насочени към конкретна демографска група или групи, сформирани на база общ интерес. Социалните мрежи се превръщат в неразделна част от ежедневието на хората, предоставяйки им начин да се свързват с другите и да бъдат информирани за света около тях.

Според проучване на GlobalWebIndex, 32% от глобалните интернет потребители на възраст 16-64 години използват социални медии, за да проучват и планират пътуването си. Този дипломен проект има за цел да предостави платформа на тези, които имат интерес към пътуването – хора, които прекарват ваканции, релаксирайки в далечни дестинации, такива, които предпочитат туризма в България и хората, които искат да научат повече за следващото си пътешествие и искат съвети от други потребители.

Основното предназначение на проектът е да позволи на хората да споделят своите пътешествия в уеб-базирано приложение. След като се регистрират, те могат да публикуват пътуванията си. Получават възможност да намерят свои приятели и да ги последват. Публикациите могат да бъдат коментирани. Всяка публикация е свързана с дестинация – мястото, където е бил публикувалият. Дестинациите могат да получават ревюта с брой звезди – от 1 до 5, заглавие и съдържание на ревюто. Потребителите могат да получават, изпращат и трият заявки за следване.

Уеб приложението организира информацията, която хората споделят относно пътуванията си. Систематизацията прави намирането ѝ от други хора по-лесно. Събирането на данни за преживяванията на хората позволява на други да се съветват с други относно следващата си екскурзия и на приятели да разкажат за пътешествията си.

Използвани термини, съкращения и абревиатури

* Уеб платформа - съвкупност от технологии, инструменти и стандарти, използвани за създаване и предоставяне на уеб базирани приложения и услуги в интернет
* Сървър – компютърна програма или хардуерно устройство, което предоставя функционалност или услуги на други програми или устройства, наречени клиенти, чрез мрежа или интернет
* Приложение
* Клас – В обектно-ориентираното програмиране (ООП) класът е план или шаблон за създаване на обекти, който определя набор от атрибути (членове на данни) и методи (функции), които обектите ще притежават. Той осигурява начин за капсулиране и организиране на свързани данни и поведение в структура за многократна употреба, която може да се инстанцира, за да се създадат отделни обекти със собствено уникално състояние и поведение
* Обект – в обектно-ориентираното програмиране (ООП) обектът е екземпляр на клас, който представлява специфична единица или концепция в програмата, състояща се от данни (свойства) и поведение (методи), които могат да бъдат манипулирани или достъпни от други обекти или части на програмата.
* API – приложно-програмен интерфейс – набор от протоколи, инструменти и стандарти за изграждане на софтуерни приложения и услуги, които могат да взаимодействат с други софтуерни приложения.
* REST – софтуерен архитектурен стил, използван за изграждане на уеб услуги, при който комуникацията между клиента и сървъра се осъществява с помощта на HTTP протоколи като GET, POST, PUT и DELETE за достъп и манипулиране на ресурси или данни.
* Entity – обект или модел на данни, който представя конкретна концепция в приложението
* Контролер – компонент на приложението, който управлява потока от данни между потребителския интерфейс и модела на данните
* Repository – компонент на приложението, който осигурява общ интерфейс за достъп и съхранение на данни, използван в слоя за достъп до данни на приложението
* Услуга (service) – компонент на приложението, който осигурява специфична функционалност или бизнес логика, често използван в обслужващия слой на приложението
* Аутентикация – процес на проверка на самоличността на потребител или система с цел осигуряване на сигурен достъп до ресурси или данни в приложението, често реализиран чрез комбинация от потребителско име/парола, токени и/или други протоколи за сигурност
* Първичен ключ – уникален идентификатор, присвоен на всеки запис (ред) в таблица от база данни, за да се осигури цялостност на данните, да се даде възможност за ефективни заявки и сортиране и да се посочат връзките с други таблици

ПЪРВА ГЛАВА

Анализ на съществуващи решения, инструменти, методи и технологии за уеб платформи за споделяне на пътувания

Концепцията за споделяне на опит от пътуване става все по-популярна през последните години, тъй като технологията улеснява и окуражава документирането и споделянето на тези преживявания онлайн. Социалните медийни платформи като Facebook, Instagram и YouTube играят значителна роля в тази тенденция, тъй като позволяват на потребителите да споделят снимки, видеоклипове и истории за своите пътувания. От друга страна, съществуват и платформи, специално насочени за за пътуване, които позволяват на потребителите да се свързват с други пътници, да споделят съвети и да планират следващите си пътувания.

* 1. Съществуващи платформи, използвани за споделяне на пътувания

В тази глава ще бъде представен анализ на някои от най-популярните налични в момента социални платформи, които могат да бъдат начин да се споделят пътувания. След това, ще бъдат съпоставени с предметът на тази дипломна работа.

* + 1. Facebook

Facebook е една от най-широко използваните социални медии в света с над 2,8 милиарда активни потребители към 2022 г. Платформата позволява на потребителите да създават личен профил, да споделят снимки, видеоклипове и истории и да се свързват с приятели и семейство . Facebook обаче не е само платформа за лични връзки, той е и популярна дестинация за споделяне на опит от пътуване.

Facebook разполага с различни функции, които го правят подходяща платформа за споделяне на личен опит и информация. Потребителите могат да създават публикации от снимки и текст и да ги споделят с приятелите и семейството си, което им позволява да документират важни събития и моменти от живота си. Друга функционалност позволява публикуването на актуализации на състоянието и да споделянето на снимки и видеоклипове в реално време. Facebook предоставя възможността потребители да се присъединяват към групи и страници, които отговарят на техните интереси, където могат да се свързват с хора с подобно мислене, да споделят съвети и да бъдат информирани за събития и събития.

Като цяло Facebook е широко използвана социална медийна платформа, която предлага разнообразни функции, чрез които хората се свързват, споделят информация и биват информирани. Той е основен инструмент, чрез който хората поддържат връзка с приятели и семейство, както и да бъдат наясно със своите интереси и света около тях.

Платформата може да бъде използвана от разнообразие от устройства, тъй като има мобилни и Desktop приложения, както и уебсайт.

* + 1. Instagram

Instagram е социална медийна платформа, базирана на визуални носачи на информация, която позволява на потребители да споделят и разглеждат снимки и видеоклипове от цял ​​свят. От стартирането си през 2010 г. тя придобива значителна популярност и нарасва до над 1 милиард активни потребители месечно към 2021 г. Платформата се превръща в популярна дестинация както за лична, така и за професионална употреба.

Instagram предлага разнообразие от функции, които улесняват споделянето на ежедневни преживявания чрез снимки и видеоклипове. Потребителите могат да създадат личен профил и да споделят съдържанието си със своите последователи, които също могат да преглеждат и да се обвързват с техните публикации. Освен това платформата предоставя инструменти за редактиране и филтри за подобряване и персонализиране на съдържанието, както и хаштагове за увеличаване на откриваемостта му.

По отношение на търговската си употреба, Instagram се превръща във важен инструмент за бизнеса, марките и влиятелните лица да се свързват с аудиторията си, да споделят своите продукти и услуги и да популяризират своята марка. Платформата предоставя разнообразие от рекламни опции и с новата функция за пазаруване се превърна в жизненоважна платформа за електронна търговия. Instagram се доказвав като ефективен начин за бизнеса да достигне до широк кръг от аудитория и да увеличи продажбите си.

* + 1. TripAdvisor

TripAdvisor е водещ уебсайт за планиране на пътувания, който позволява на потребителите да проучват и резервират своите екскурзии. Платформата е основана през 2000 г. и оттогава се превръща в един от най-големите уебсайтове за отзиви за пътувания в света с над 860 милиона отзива и мнения към 2021 г. TripAdvisor предоставя богата информация за пътниците, включително отзиви за хотели, ресторанти, и атракции, както и възможност за сравняване на цените от различни уебсайтове за хотелски резервации.

Една от ключовите характеристики на TripAdvisor е неговата изчерпателна колекция от генерирани от потребителите рецензии. Тези рецензии предоставят ценна информация и обратна връзка за пътуващите, което им позволява да вземат информирани решения относно своите планове за пътуване. Платформата обхваща широка гама от дестинации и места за настаняване, а рецензиите са написани от реални пътници, които имат личен опит от имотите, които оценяват. Потребителите могат също да споделят свои собствени отзиви и снимки от своите пътувания на платформата, предоставяйки им начин да споделят своя опит с други.

TripAdvisor предлага разнообразие от инструменти, за да помогне на потребителите да планират своите пътувания, включително функция за планиране на пътуване, която позволява на потребителите да създават и запазват маршрути, както и изглед на карта, който показва местоположението на хотели, ресторанти и атракции. Платформата също така предлага разнообразие от филтри, като ценови диапазон и местоположение, които могат да помогнат на потребителите да намерят идеалното настаняване за техния бюджет и нужди.

TripAdvisor има и функция за резервация, която позволява на потребителите да резервират своето настаняване, полети и дейности директно в платформата. Тази функция предоставя удобно обслужване на едно място, за да бъде организирано цялото пътуване. TripAdvisor има мобилно приложение за устройства с iOS и Android.

* + 1. Booking.com

Booking.com е водеща онлайн туристическа агенция, чрез която хора търсят и резервират настанявания, полети и коли под наем. Платформата е основана през 1996 г. и оттогава е една от най-големите онлайн туристически агенции в света, с над 28 милиона имота, изброени на нейния уебсайт към 2022 г. Booking.com предлага широка селекция от места за настаняване, включително хотели, курорти, ваканционни жилища под наем и хостели, което го прави едно гише за всички нужди, свързани с пътуване.

Една от ключовите характеристики на Booking.com е огромната колекция от опции за настаняване. Платформата обхваща широка гама от дестинации, а имотите, изброени на платформата, са разнообразни, вариращи от бюджетни опции до луксозни хотели. Това разнообразие от опции улеснява потребителите да намерят идеалното настаняване, което отговаря на техния бюджет и нужди. Освен това Booking.com предлага и различни филтри, като ценови диапазон, местоположение и удобства, които могат да помогнат на потребителите да стеснят търсенето си и да намерят идеалното настаняване за своето пътуване.

Booking.com също предлага различни инструменти, които улесняват планирането и резервацията на пътувания. Търсачката на платформата позволява на потребителите да намират и сравняват цените от различни имоти, а удобният за потребителя интерфейс на платформата улеснява потребителите да резервират своето настаняване, полети и коли под наем. Платформата предлага и мобилно приложение за iOS и Android устройства, което позволява на потребителите да имат достъп до функциите на платформата и резервациите в движение.

Booking.com има различни опции за плащане, включително кредитни карти, дебитни карти и PayPal. Платформата също предлага различни политики за анулиране и възстановяване на суми, които предоставят на потребителите гъвкавост и спокойствие при планирането на пътуванията си. Освен това Booking.com предлага и програма за лоялност Genius, която предоставя на потребителите изключителни оферти и отстъпки за техните резервации.

* + 1. Foursquare City Guide

Foursquare е уеб-базирано и мобилно приложение, което позволява на потребителите да споделят местоположението си, да откриват нови места и да се свързват с други. Платформата стартира през 2009 г. и оттогава се превърна във водеща платформа, с над 50 милиона активни потребители месечно към 2021 г. Foursquare предоставя персонализирано изживяване за потребителите чрез своята комбинация от алгоритми за машинно обучение и човешко управление.

Една от ключовите характеристики на Foursquare е способността му да предоставя персонализирани препоръки на потребителите въз основа на тяхното местоположение и минали взаимодействия. Платформата използва комбинация от алгоритми за машинно обучение и човешко управление, за да предложи персонализирани предложения за близки места, като ресторанти, кафенета, барове и магазини. Освен това Foursquare позволява на потребителите да запазват и споделят предпочитаните от тях места, а също така могат да следват други потребители, за да видят техните препоръки.

Foursquare така предлага разнообразие от инструменти, за да помогне на потребителите да открият нови места и да се свържат с хора с подобно мислене. Функцията за търсене на платформата позволява на потребителите да намират и изследват места въз основа на тяхното местоположение. Освен това мобилното приложение на платформата за устройства с iOS и Android позволява на потребителите достъп до функциите и препоръките на платформата в движение.

Foursquare също така предлага набор от функции за бизнеса, което им позволява да се свързват с клиентите си, да споделят информация за своите продукти и услуги и да популяризират своята марка. Функцията за рекламиране на базата на местоположение на платформата позволява на бизнеса да насочва рекламите си към потребители в определени области, а функцията за анализ на платформата предоставя на бизнеса ценни данни за поведението и предпочитанията на клиентите.

* + 1. Tumblr

Tumblr е уебсайт за блогове, който позволява на потребителите да споделят текст, снимки, видеоклипове и линкове. Платформата стартира през 2007 г. и оттогава се превръща във водеща платформа за микроблогинг с над 446 милиона блога към 2022 г. Tumblr предоставя лесна за използване платформа за споделяне на своите мисли, опит, цитати и интереси с други.

Най-популярната функционалност на Tumblr е създаването и персонализирането на блогове в няколко клика. Удобният за потребителя интерфейс на платформата и простите инструменти за блогове улесняват потребителите да създават и персонализират своите блогове. Освен това Tumblr позволява на потребителите да следват блоговете на други потребители, което им позволява да откриват нови блогове и да са в крак с най-новите публикации.

Tumblr също предлага разнообразие от функции, които улесняват потребителите да споделят и откриват ново съдържание. Системата за маркиране на платформата позволява на потребителите лесно да намират и откриват нови публикации въз основа на конкретни теми, а функцията за повторно блогване на платформата позволява на потребителите лесно да споделят публикации от блогове на други потребители. Освен това Tumblr предлага и мобилно приложение за iOS и Android устройства, което позволява на потребителите да имат достъп до функциите на платформата и блоговете в движение.

Tumblr също предлага разнообразие от функции за фирми и марки. Рекламната функция на платформата позволява на бизнеса да насочва рекламите си към специфични аудитории, а функцията за анализ на платформата предоставя на бизнеса ценна информация за поведението и предпочитанията на аудиторията. Освен това Tumblr предлага разнообразие от инструменти, които позволяват на фирми и марки да популяризират своите продукти и услуги и да се свързват с аудиторията си.

* + 1. Pinterest

Pinterest е визуално базирана социална медийна платформа, която позволява на потребителите да откриват и запазват идеи за различни теми като домашен декор, мода, готвене и пътуване. Платформата стартира през 2010 г. и оттогава се превръща във водеща визуална търсачка с над 459 милиона активни потребители месечно към 2021 г. Pinterest предоставя уникално и персонализирано изживяване за потребителите чрез своите функции за визуално търсене и откриване.

Една от ключовите характеристики на Pinterest е неговата функция за визуално търсене и откриване. Платформата позволява на потребителите да откриват нови идеи и вдъхновение чрез различни пинове, дъски и теми. Потребителите могат също така да запазват, организират и споделят пинове, които харесват, като им предоставят персонализирано изживяване. Платформата също така позволява на потребителите да търсят идеи и вдъхновение по ключови думи, което улеснява намирането на конкретни теми или интереси.

Pinterest също предлага разнообразие от инструменти, които да помогнат на потребителите при организирането и споделянето на техните пинове. Функцията за дъски на платформата позволява на потребителите да организират своите щифтове в различни категории, което ги прави лесни за намиране на щифтовете, които търсят. Освен това функцията за споделяне на платформата позволява на потребителите да споделят своите пинове с други, а мобилното приложение на платформата за устройства с iOS и Android позволява на потребителите достъп до функциите на платформата и пинове в движение.

Pinterest също предлага разнообразие от функции за фирми и марки. Рекламната функция на платформата позволява на бизнеса да насочва рекламите си към специфични аудитории, а функцията за анализ на платформата предоставя на бизнеса ценна информация за поведението и предпочитанията на аудиторията. Освен това Pinterest предлага разнообразие от инструменти, които позволяват на фирмите и марките да популяризират своите продукти и услуги и да се свързват с аудиторията си.

* + 1. Сравнение

Докато разгледаните платформи предоставят възможността за регистрация, интеракция с други потребители, публикация на снимки и текст, основните им цели не разрешават проблематиката, с която този проект се занимава. Фокусът на дипломния проект е да колекционира и презентира съдържание, свързано с пътуване на различни дестинации. Това включва реален, личен опит и преживявания, които могат да упътят други, да вдъхновят или да информират в сферата на туризма.

* 1. Известни езици и среди за разработка
     1. Езици за разработка на сървърна част

Езиците, използвани за разработка на сървърни системи, отговарят за функционалността от страна на сървъра, която включва работа с данни, обработка и обслужване на заявки, направени от потребителския интерфейс.

* + - 1. Python

Python е език от високо ниво, който е известен със своята четливост и гъвкавост. Разполага с голям брой библиотеки и софтуерни рамки (frameworks), като Django и Flask. Python се използва и за научни изчисления, анализ на данни, машинно обучение и разработване на уеб сайтове, което го прави универсален език за широк спектър от приложения.

* + - 1. Java

Java е обектно-ориентиран език за програмиране със статично типизиране, който се използва активно повече от две десетилетия. Java е популярен за разработване на широкомащабни корпоративни приложения, тъй като е високо мащабируем и разполага с богата библиотека от инструменти и програмни интерфейси (API-та). Java също така има голяма и активна общност, която предоставя богата подкрепа и ресурси на програмистите. Java е подходяща за използване в много различни видове приложения, включително уеб, мобилни, настолни и сървърни приложения.

* + - 1. Kotlin

Kotlin е модерен, междуплатформен език за програмиране със статично типизиране, който работи с виртуалната машина Java. Той е проектиран така, че да бъде напълно оперативно съвместим с Java и да предоставя по-кратък и изразителен синтаксис за разработване на приложения. Kotlin набра популярност заради краткия си и изразителен синтаксис, който улеснява разработчиците да пишат поддържан и четим код. Той се използва и за разработване на приложения за Android, което го прави популярен вариант за мобилна разработка. Освен това Kotlin има разрастваща се общност и редица от инструменти и библиотеки, които могат да се използват за създаване на висококачествени приложения.

* + 1. Езици за разработка на потребителски интерфейс
       1. Традиционен HTML/CSS/JavaScript

Този метод включва писане на HTML код за създаване на структурата на потребителския интерфейс, CSS код за оформяне на потребителския интерфейс и JavaScript код за добавяне на интерактивност и динамично поведение. Този метод осигурява висока степен на контрол върху външния вид и поведението на потребителския интерфейс, но също така може да отнеме много време и да бъде сложен, особено за по-мащабни приложения.

* + - 1. Angular

Angular е популярен способ с отворен код за потребителски интерфейс, разработена от Google. Той е базирана на TypeScript, статично въведен език за програмиране, и осигурява високо ниво на абстракция за изграждане на сложни потребителски интерфейси. Angular използва архитектура, съставена от компоненти, което позволява на разработчиците да изграждат компоненти на потребителския интерфейс за многократна употреба, които лесно могат да бъдат сглобени в цялостно приложение. Освен това Angular предоставя широк набор от инструменти и функции, като двупосочно свързване на данни, инжектиране на зависимости и обширна библиотека от компоненти на потребителския интерфейс, което го прави цялостно решение за изграждане на сложни уеб приложения.

* + - 1. React

React е библиотека на JavaScript за изграждане на потребителски интерфейси. Тя е разработена и се поддържа от Facebook и се използва за създаване на уеб приложения, мобилни приложения и приложения за компютри. React използва виртуален DOM (Document Object Model) за ефективно актуализиране на потребителския интерфейс, което го прави бърз и отзивчив дори за сложни потребителски интерфейси. Компонентите на React могат да се съставят по модулен начин, което позволява на разработчиците да изграждат сложни потребителски интерфейси чрез комбиниране на прости компоненти. Освен това React предоставя редица инструменти и функции, като JSX, синтактично разширение за JavaScript, и голяма екосистема от библиотеки и инструменти, което го прави популярен избор за изграждане на уеб приложения.

* + - 1. Vue

Vue.js е прогресивна софтуерна рамка за изграждане на потребителски интерфейси. Тя е проектирана така, че да бъде лека и бърза, което я прави добър избор за изграждане както на прости, така и на сложни потребителски интерфейси. Vue.js използва архитектура, базирана на компоненти, което улеснява изграждането на компоненти на потребителския интерфейс за многократна употреба. Освен това Vue.js предоставя редица функции, като двупосочно свързване на данни, реактивен модел на данните и модулна архитектура, което го прави гъвкаво и мащабируемо решение за изграждане на уеб приложения. Освен това Vue.js разполага с нарастваща екосистема от плъгини, библиотеки и инструменти, което го прави популярен избор за изграждане на уеб приложения.

* + 1. Бази данни
       1. Релационни

Релационните бази данни са вид система за управление на бази данни, която съхранява данни в таблици, съставени от редове и колони. Колоните представляват различните атрибути на данните, а редовете - отделните записи. Връзките между данните се установяват чрез използването на ключове, които свързват данните в една таблица със съответните данни в друга. Тази структура позволява лесно извличане и манипулиране на данните, както и налагане на целостта на данните чрез използване на ограничения и връзки. Релационните бази данни се използват масово за различни приложения - от малки персонални бази данни до големи корпоративни системи - поради тяхната мащабируемост, надеждност и лекота на използване.

* + - * 1. MySQL

MySQL е система за управление на релационни бази данни с отворен код, която се използва активно при създаването на уеб приложения и други приложения. Известна е със своята надеждност, бърза работа и лекота на използване. MySQL може да обработва големи обеми от данни, което я прави популярен избор за уебсайтове, които изискват много място за съхранение. Освен това е силно адаптивен, като за подобряване на функционалността му са налични голям брой приставки и разширения.

* + - * 1. Oracle Database

Oracle Database е разработена от Oracle Corporation и е известна със своята мащабируемост, надеждност и висока производителност, което я прави популярен избор за приложения. Oracle Database предлага и редица усъвършенствани функции, като например поддръжка на големи обеми от данни, изчисления в облак и изчисления в паметта. Тя се използва широко във финансовия и правителствения сектор и др.

* + - * 1. PostgreSQL

PostgreSQL е релационна система, която работи с отворен код и поддържа различни типове данни. Освен това тя е силно приспособима, с голям брой налични разширения за добавяне на функционалност. PostgreSQL често се използва за уеб приложения, складове за данни и геопространствени приложения, наред с други. Известен е със силната си поддръжка на SQL стандартите, което го прави популярен избор за тези, които търсят релационна база данни с голям брой функции.

* + - 1. Нерелационни

Нерелационните бази данни, известни също като NoSQL бази данни, са вид системи за управление на бази данни, които се различават от традиционните релационни бази данни по подхода си към съхранението и управлението на данни. За разлика от релационните бази данни, които използват таблици и редове за организиране на данните, NoSQL базите данни използват различни структури от данни, като документ, чифт от клю и стойност или графики, за съхранение и управление на данни. Това им позволява да бъдат по-гъвкави и мащабируеми, което ги прави подходящи за съвременни приложения, изискващи висока производителност, мащабируемост и възможност за обработка на големи количества неструктурирани данни. NoSQL базите данни често са проектирани да бъдат разпределени и децентрализирани, което ги прави достъпни и устойчиви. Благодарение на тези характеристики NoSQL базите данни стават все по-популярни за широк спектър от приложения, включително за анализ на данни в реално време, уеб, мобилни и IoT приложения и много други.

* + - * 1. Redis

Redis е хранилище за структури от данни, което може да се използва като база данни, кеш и брокер на съобщения. Той е известен с високата си производителност и мащабируемост, което го прави популярен избор за анализ на данни в реално време, игри и други приложения, които изискват бърз достъп до данни. Redis поддържа разнообразни структури от данни, включително низове, хешове, списъци и множества, което го прави изключително гъвкав. Освен това той е силно адаптивен, като са налични разнообразни опции за добавяне на функционалност и адаптиране на услугата към конкретни нужди.

* + - * 1. MongoDB

MongoDB е популярна NoSQL база данни, която е характерна с лесната си употреба, мащабируемост и поддръжка на съхранение на данни на базата на документи. Тя се използва активно за уеб приложения, системи за управление на съдържанието и за анализ на данни в реално време. MongoDB също така е силно приспособима, като за добавяне на функционалност са налични голям брой приставки и разширения. Той е проектиран да бъде силно мащабируем, с възможност за обработка на големи количества данни и поддръжка на хоризонтално мащабиране.

* + 1. Среди и инструменти за разработка
       1. InteliJ

IntelliJ IDEA е интегрирана среда за разработка на Java (IDE), разработена от JetBrains. Тя предоставя широк набор от функции, включително завършване на кода, проверка на кода и интеграция с контрола на версиите. IntelliJ IDEA е известна със силно персонализирания си интерфейс и поддръжката на голям брой езици за програмиране, което я прави идеален избор за разработчици, работещи по сложни проекти. Той също така разполага със стабилна екосистема от плъгини, която позволява на разработчиците да добавят нови функции и да разширяват функционалността му.

* + - 1. VSCode

Visual Studio Code (VSCode) е популярен и ефикасен редактор на код, разработен от Microsoft. Той е междуплатформен и поддържа широк набор от езици за програмиране и предоставя функции като отстраняване на грешки, интегриран терминал и навигация в кода. VSCode също така е персонализируем, разполагайки с голям пазар за разширения и теми, които могат да се използват за подобряване на неговата функционалност. Той има прост и интуитивен потребителски интерфейс, което го прави идеален избор както за начинаещи, така и за опитни разработчици.

* + - 1. MySQL Workbench

MySQL Workbench е графичен инструмент за управление и проектиране на бази данни. Той предоставя визуален интерфейс за създаване, изпълнение и оптимизиране на SQL заявки и позволява на разработчиците да управляват своите бази данни чрез графичен интерфейс. MySQL Workbench включва и функции като моделиране на данни, генериране на SQL код и миграция на бази данни, което го прави универсално решение за управление на бази данни. Той е наличен за множество операционни системи и се използва широко от администратори на бази данни и разработчици за управление на техните бази данни MySQL.

ВТОРА ГЛАВА

Архитектура на уеб платформа за споделяне на пътувания

* 1. Изисквания към програмния продукт
     1. Аутентикация и регистрация на потребителя

Преди да започне да използва платформата, човекът трябва да се регистрира, или, ако вече има профил, да влезе в него чрез електронна поща или потребителско име и парола, създадена от него при регистрация. Ако потребител е влязъл в профила си, той може да излезе от него.

* + 1. Персонализация на профил

В профила могат да бъдат поставени: пол, описание на профила или кратко въведение, местоживеене.

* + 1. Поверителност

По желание и предпочитание на потребителя, той може да направи профила си заключен. По този начин само хора, които са получили одобрена от него заявка за следване, могат да разглеждат неговите пътешествия. Освен това, може да променя поверителността на пътешествията си, така че да бъдат виждани само от него, само от приятелите му, само от последвалите го, или публично.

* + 1. Създаване на пътешествия (journeys) и свързаните с тях дейности (activities)

При създаването на пътешествия, потребителят може да постави дати на начало и край и описание. Трябва да избере дестинация

* + 1. Следване на хора и създаване на приятелства

Един потребител може да следва хора, за да вижда техните пътешествия на началната страница на уеб приложението. Ако профилът на този, когото потребителят иска да последва е заключен, то потребителя може да изпрати заявка за следване, която може да бъде одобрена или отхвърлена.

* + 1. Разглеждане на пътешествия и получаване на персонализирано препоръчани пътешествия

На началната страница могат да бъдат видени препоръчани, на база на интереси, пътешествия.

* + 1. Поставяне на коментари на пътешествия

Пътешествията могат да бъдат коментирани като коментарите играят роля в алгоритъма за препоръчани дестинации и пътешествия.

* + 1. Получаване на персонализирани предложения за дестинации

Продуктът пресмята интересът на потребителят към дадени характеристики и препоръчва дестинации с тези характеристики.

* + 1. Поставяне на ревю на дестинации

Потребителят може да постави ревю на дестинация по негов избор. Ревюто може да бъде озаглавено, да получи текстово съдържание и оценка от 1 до 5 звезди. Ревютата играят роля в пресмятането на средната оценка на дестинация. От своя страна, ревютата могат да бъдат изтрити от създателя им.

* 1. Избрани езици и инструменти за програмиране
     1. Езици и инструменти, използвани в сървърната част
        1. InteliJ Idea Ultimate

Сървърната част на платформата е написана с IntelliJ Ultimate – интегрирана среда за разработка (IDE), която поддържа Spring Boot. IntelliJ Ultimate предоставя цялостен набор от инструменти и функции, които улесняват разработването, тестването и внедряването на приложения Spring Boot, включително вградена поддръжка за ключови технологии като Maven.

* + - 1. Kotlin

Kotlin е език за програмиране, разработен от JetBrains, който е напълно съвместим с Java и може да се използва за разработване на широка гама от приложения, включително мобилни, уеб и бекенд. Известно е, че е по-изразителен, разбираем и безопасен от Java. Освен това има функции като нулева безопасност и структури от данни (класове), които улесняват писането на четим, поддържаем и ефективен код. Освен това Kotlin има нарастваща общност и много библиотеки и софтуерни рамки, които могат да се използват за изграждане на сървъра.

* + - 1. Spring Boot

Избраната софтуерна рамка за сървърната част е Spring Boot. Spring Boot e софтуерна рамка за Java и Kotlin, чрез която се разработват приложно-програмни интерфейси. Тя предлага няколко предимства, като например лесно конфигуриране, автоматично конфигуриране и инжектиране на зависимости (dependency injection). Spring Boot работи с анотации. Анотациите се използват за предоставяне на метаданни на Spring Framework, което позволява изпълняването различни задачи, като инжектиране на зависимости, аспектно-ориентирано програмиране и уеб конфигурация. Анотациите се използват често в Spring, за да се намали броят на шаблонния код и да се улесни конфигурирането и управлението на сложни приложения.

* + - 1. REST архитектура

Със Spring Boot e реализиран приложно-програмен интерфейс, имплементирал REST архитектурата. REST (Representational State Transfer) е архитектурен стил за изграждане на уеб услуги, който позволява обмен на данни между приложения. Това е набор от ограничения и принципи, които определят как трябва да се проектират и реализират уеб услуги, което улеснява изграждането на мащабируеми, надеждни и поддържани уеб услуги.

Архитектурата REST използва модела клиент-сървър, при който клиентът изпраща заявки до сървъра, а сървърът изпраща отговори обратно на клиента. Комуникацията между клиента и сървъра е без състояние (stateless), което означава, че сървърът не съхранява информация за състоянието на клиента. Вместо това всяка заявка от клиента съдържа цялата информация, необходима за изпълнение на заявката.

REST уеб услугите (наричани още RESTful услуги) са проектирани да бъдат ресурсно ориентирани, което означава, че ресурсите се идентифицират чрез уникални URI (Uniform Resource Identifiers). Всеки ресурс може да бъде достъпен, манипулиран и представен по различни начини, като се използват методите на HTTP (Hypertext Transfer Protocol), като GET, POST, PUT и DELETE, използвани съответно за получаване, изпращане, модификация или триене на информация.

Използването на архитектурата REST за уеб услуги има няколко предимства. Първо, RESTful услугите са прости и лесни за разбиране, което ги прави идеални за бързо изграждане на уеб приложения. Второ, те са силно мащабируеми и могат да обработват голям брой заявки от клиенти. Трето, архитектурата REST е независима от езика, което я прави лесна за интегриране с различни програмни езици и технологии. И накрая, тя е гъвкава и може да поддържа различни формати на данни, като JSON (JavaScript Object Notation) и XML (Extensible Markup Language).

Моделът клиент-сървър и комуникацията без състояние гарантират, че платформата може да обработва голям брой заявки от потребители, а ресурсно ориентираният дизайн позволява лесно манипулиране и представяне на данни с помощта на HTTP методи. Освен това архитектурата REST осигурява стандартизиран начин за интегриране на различни програмни езици и технологии, което улеснява разработването и поддръжката на платформата.

* + - 1. Maven

Maven е инструмент за автоматизиране на изграждането на проекти, базирани на Java, който предоставя инструмент за управление на зависимостите на проекта, генериране на изпълними файлове и автоматизиране на процеса на изграждане. Той се основава на декларативен конфигурационен файл, базиран на XML, наречен pom.xml, който съдържа настройките за изграждане на проекта, зависимостите и библиотеките.

Maven предоставя редица предимства, включително стандартизиран процес на изграждане, централизирано управление на зависимостите на проекта и подобрена поддържаемост на проекта. Той осигурява и поддръжка на непрекъсната интеграция, което позволява на разработчиците да автоматизират изграждането, тестването и внедряването на своите приложения.

* + - 1. Vader

VADER (Valence Aware Dictionary and sEntiment Reasoner) е инструмент за анализ на сентимента, с други думи – настроенията, предназначен за използване със съдържание от социални медии. Той използва лексикон от думи и фрази, свързани с настроенията, заедно с граматични правила и синтактични конвенции, за да определи цялостното настроение на даден текст.

VADER предоставя редица предимства в сравнение с традиционните инструменти за анализ на настроенията, включително подобрена точност, възможност за работа с емотикони и други нестандартни текстови характеристики, както и възможност за идентифициране на нюансирани изрази на настроения, включително и ирония. Освен това той е с отворен код, което го прави достъпен за широк кръг потребители и разработчици.

В проектът на тази дипломна работа, VADER се използва за анализ на генерирани от потребителите данни, като например коментари на пътешествия. След това тази информация се използва за подобряване на потребителското изживяване, като се предлага съобразено с интересите на потребителите съдържание.

* + 1. Система за управление на базата данни

За да се запазва информацията за потребителите, пътешествията им, дестинациите и др. Е използвана MySQL база данни. MySQL предоставя интуитивен начин да бъдат менажирани данни. MySQL е релационна система за управление на бази данни, което означава, че съхранява данни в таблици с предварително определени връзки. Това улеснява организирането и управлението на големи обеми от данни, особено когато става въпрос за сложни взаимоотношения между различни единици данни. MySQL е известен с високата си производителност и мащабируемост, което го прави подходяща за обработка на големи количества данни и тежък трафик. Има отлични възможности за четене и писане и може ефективно да обработва множество заявки едновременно. MySQL е съвместим с различни операционни системи и езици за програмиране, което го прави лесен за интегриране с различни уеб ехнологи и приложения.

* + 1. Софтуерна рамка за реализацията на потребителския интерфейс

Nuxt.js в комбинация с Vue.js 3 е избраният инструмент за разработка на потребителския интерфейс, поради възможностите му за визуализация от страна на сървъра, модулната архитектура и поддръжката на статично генериране на сайтове. Тези функции допринасят към производителността и потребителското изживяване на уеб приложението, особено при потребители с по-бавни интернет връзки или по-стари устройства.

Предоставяйки възможност за рендиране от страна на сървъра (SSR – server side rendering), Nuxt.js може да възпроизвежда страници, преди да ги изпрати на клиента, което може да намали натоварването на клиентското устройство и да подобри цялостното потребителско изживяване. Модулната архитектура на Nuxt.js позволява лесно персонализиране и подобрена функционалност, а поддръжката му за генериране на статични сайтове може да помогне за подобряване на мащабируемостта и производителността на приложението.

Освен това Nuxt.js предоставя няколко други функции, включително автоматично разделяне на кода, кеширане и оптимизирана доставка на активи, които могат допълнително да подобрят производителността и потребителското изживяване на приложението. Чрез Nuxt.js с Vue.js 3 платформата се възползва от мощна комбинация от инструменти, които могат да помогнат за подобряване на производителността, скалируемостта и персонализирането на уеб приложението, и не на последно място, значително подобрява изживяването на програмиста (developer experience – термин, използван за описание на степента на оптимизация на процеса на програмиране).

* + - 1. Tailwind CSS

Tailwind CSS е софтуерна рамка, която позволява на програмистите бързо да създават потребителски интерфейси. С изчерпателен набор от предварително изградени CSS класове Tailwind CSS оптимизира процеса на стилизиране и улеснява създаването на адаптивни оформления за различни устройства. Нейната простота и гъвкавост позволяват на разработчиците да проектират с лекота модерни и уеб приложения. Освен това поддръжката на рамката за персонализирани конфигурации и разширения я прави универсален инструмент за различни проекти за разработка.

* + - 1. Axios

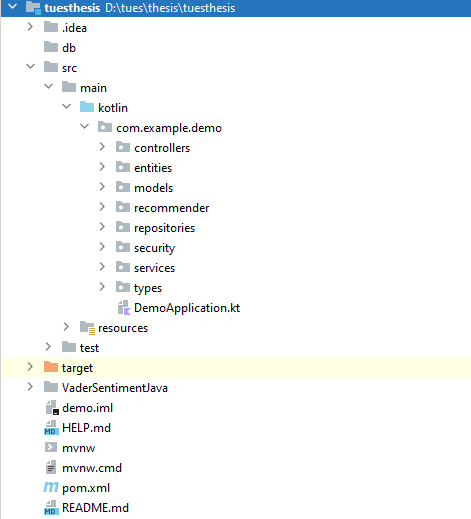
Axios е популярна библиотека на JavaScript, която се използва за извършване на HTTP заявки от уеб браузъри и Node.js приложения. Тя предоставя прост и интуитивен API за извършване на обичайни HTTP операции, като GET, POST, PUT, DELETE и PATCH, и поддържа функции като прихващане на заявки и отговори, отмяна и обработка на грешки. Axios също така поддържа автоматична сериализация и десериализация на JSON данни и може лесно да се интегрира със съвременни JavaScript софтуерни рамки.

* + - 1. Pinia

Pinia е библиотека за за Vue.js, която предоставя прост и скалируем начин за управление на състоянието на приложенията по реактивен и композиционен начин. Pinia е известна с лекия си и гъвкав API, който улеснява създаването и управлението на хранилища на състояния, които могат лесно да се интегрират с компоненти на Vue.js и други библиотеки. Освен това Pinia поддържа усъвършенствани функции, като например типово сигурен достъп до състоянието, проследяване на реактивността и поддръжка на плъгини, което я прави популярен избор за разработчиците, изграждащи Vue.js приложения.

* 1. Описание на алгоритъма
     1. Архитектура на приложно-програмния интерфейс
        1. Файлова структура

На фигура 2.1 е илюстрирана файловата йерархия на сървърната част на проекта.

Всяка папка в папката „kotlin” се нарича пакет. В Kotlin пакетът е механизъм за организиране на свързани класове и функции в пространство от имена. Чрез групиране на свързан код в пакет кода става модуларен, може да се използва многократно и да се поддържа по-лесно. Пакетите в Kotlin могат да бъдат вложени, което позволява създаването на йерархични структури от свързан код.

В папката „target” се съхранява компилираният байткод на приложението след процеса на компилиране. След това компилираният байткод може да бъде стартиран на Java виртуалната машина (JVM), за да се изпълни приложението.

Всеки файл от пакета „com” съдържа по един клас, наричан още обект. Класът е основната структурна единица на обектно-ориентираните езици като Kotlin.

*Фигура 2.1 – Файлова структура*

Папката „resources” е специална папка, използвана за съхранение на ресурси, които не са свързани с Kotlin, като например конфигурационни файлове, файлове с properties – свойства, които се използват от приложението по време на изпълнение. Тези ресурси могат да бъдат достъпни за приложението чрез използване на релативни пътища към класа, което улеснява организирането и управлението на ресурсите на приложението.

Файлът „demo.iml” е файл на модул на IntelliJ IDEA, който съдържа конфигурацията и настройките за конкретен модул в рамките на проекта. Той включва информация като име на модула, изходна и изходна директория, зависимости и настройки за компилация. Файлът .iml обикновено се използва за съхраняване на специфична за проекта информация, която не е включена в скриптовете за изграждане на проекта, и се използва от IntelliJ IDEA за управление на конфигурацията и поведението на модула.

Файлът „README.md” е документ, който обикновено придружава даден софтуерен проект и предоставя информация за това как да бъде използван и/или инсталиран проекта.

* + - 1. Основни обекти

В Spring обектите, наречени „entities”, се използват за представяне на бизнес обекти и се съпоставят с таблици от бази данни. Spring предоставя няколко инструмента за работа с „entity“ обекти, като Spring Data JPA, която позволява лесното извършване на общи CRUD (създаване, четене, актуализиране и изтриване) операции върху обектите, както и по-усъвършенствани операции, като например създаване на заявки и страници от данни (pagination). Тези обекти се съдържат в пакета „entities”.

* + - * 1. Потребител – UserEntity

Обектът за потребител представлява регистриран в приложението потребител и съдържа основната за дадения човек информация. Тази информация се използва за улесняване на взаимодействието на потребителя с приложението, включително възможността да търси и да се присъединява към пътувания, да публикува отзиви и да общува с други потребители. Този клас е съществен компонент в функционалността на платформата.

* + - * 1. Детайли на потребителя – UserDetailsEntity

Обектът за детайли на потребителя е компонент на същността „Потребител“, който съдържа допълнителна информация за регистрирания потребител, като дата на раждане, пол, местожителство. Тази информация се използва за осигуряване на по-пълен профил на потребителя, който може да се използва за свързване на хора със сходни интереси и предпочитания и за осигуряване на по-персонализирано преживяване за всеки потребител.

* + - * 1. Дестинация – DestinationEntity

Това е обектът, който представлява желаната географското място на пътуването, посочена от потребителя при подаване на заявка за пътуване.

* + - * 1. Код на характеристика на дестинация – FeatureCod

Характеристика на DestinationEntity, която дава повече информация за дестинацията.

* + - * 1. Държава – CountryEntity

Държава, в която се намира дестинацията. Значителна част от дестинациите са свързани с държави, но някои не принадлежат на такива.

* + - * 1. Ревю на дестинация – ReviewEntity

Потребителите могат да публикуват мнения и впечатления от определена дестинация, като публикуват ревюта.

* + - * 1. Пътешествие – JourneyEntity

Обектът за пътешествие е един от основните елементи на уеб платформата. Чрез него потребителят описва свое пътешествие.

* + - * 1. Дейност по време на пътешествие – ActivityEntity

Този обект представлява дейност, извършена по време на пътешествието и повишава нивото на интерактивност и изразителност на обекта за пътешествие.

* + - * 1. Коментар на пътешествие – CommentEntity

Коментарите се публикуват от потребители под пътешествия и спомагат подобряването на потребителското изживяване, като дават база, на която могат да се преценят интересите на потребителя, който публикува коментара.

* + - * 1. Интерес към дадени характеристики на пътешествия – InterestEntity

Обектът за интерес основава подбирането на препоръчани за даден потребител пътешествия. Например, ако има данни, че определен потребител има позитивно настроение към държавата България, то ще бъдат препоръчани пътешествия, чиито дестинации се намират в България.

* + - * 1. Известие – NotificationEntity

Целта на обектите за известия е да уведомяват потребителя за обвързване на други хора с негово съдържание в кратко изречение.

* + - * 1. Следване – FollowEntity

FollowEntity e клас, който дава сведения за следванията между потребители.

* + - * 1. Заявка за следване – FollowRequestEntity

Този клас е част от функционалността на приложението. Предлагаща поверителност. При поверителни профили, преди да се създаде FollowEntity, се създава FollowRequestEntity, което позволява на потребителя да одобри следването на някой друг или да изтрие заявката, с което той отказва следването.

* + - 1. Контролери

Контролерите са ключов компонент на REST архитектурата, който позволява на приложението да получава и отговаря на входящи заявки от потребителите. Контролерите отговарят за съпоставянето на входящите заявки към конкретни методи и услуги в приложението и за връщането на подходящи отговори въз основа на вида на заявката. Служейки като мост между компонентите на уеб страницата и приложно-програмния интерфейс на приложението, контролерите играят важна роля за улесняване на взаимодействието с потребителите и осигуряване на безпроблемно потребителско изживяване.

Всеки контролер в приложението представлява клас, като в параметрите си има поне по един (обикновено един) клас за услуги, който отговаря за отговорите на заявките, приети от контролера.

* + - 1. Услуги

Услугите, наричани още „сървиси“ (от services – услуги) отговарят за капсулирането и управлението на бизнес логиката на приложението, която може да варира от управление на потребителски акаунти и предпочитания до обработка на заявки за пътуване и комуникация между потребители.

* + 1. Проектиране и създаване на база данни
       1. Таблици

Всеки клас от точка 2.3.1.2 се запазва в отделна таблица.

* + - * 1. users

Таблицата „users” съдържа всички потребители. С други думи, един ред / запис от таблицата представлява едно UserEntity. Тя има следните колони:

* id – първичен ключ
* username – потребителско име
* email – електронна поща
* password – парола
* details\_id – външен ключ към реда от details таблицата, който се отнася за потребителя
* first\_name – първото име на потребителя

На фигура 2.2 е MySQL скриптът, с който се създава таблицата users.

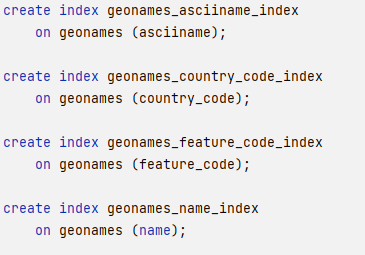
* last\_name – фамилия

Фигура 2.2 – MySQL скрипт за създаване на таблицата users

* + - * 1. user\_details – съхранява UserDetailsEntity-та
* id – първичен ключ
* biography – кратко описание за потребителя, въведено от него.
* gender – пол на потребителя
* registration\_timestamp – точен час на регистрация
* birthdate – рожденна дата
* privacy – поверителност на профила
* residence - местожителство
  + - * 1. geonames - съхранява DestinationEntity

Фигура 2.3 – MySQL скрипт за създаване на таблицата user\_details

„geonames“ е таблицата, пазеща информация за дестинациите и всеки един нейн ред съответства на едно DestinationEntity (точка 2.3.1.2.3). За да се запълни тази таблица с данни за местоположения, e използван приложният програмен интерфейс end[[1]](#endnote-1). Нейните колони са:

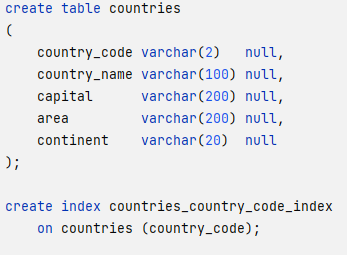
* geonameid – Основният идентификатор на ред от таблицата, с други думи на дестинация
* name – име на дестинацията
* asciiname – име на дестинацията, обозначено с ASCII символи
* alternatenames – други имена, с които дестинацията е позната
* latitude – географска ширина
* longitude – географска дължина
* feature\_class – еднобуквен код, с който се характеризира класът на дестинацията – например „A“ е за държава, щат или регион
* feature\_code – код, характеризиращ вида на дестинацията, напр. „RF“ обозначава рифове
* country\_code – двубуквен код на държавата, в която се намира дестинацията
* cc2, admin1 code, admin2 code, admin3 code, population, elevation, dem, timezone, modification date – Това са колони, които не се използват в приложението, но въпреки това, не трябва да бъдат изтрити или модифицирани, както е указано в документацията на geonames[[2]](#endnote-2)

Фигура 2.5 – Индекси върху колони на geonames, създадени за по-бързо навигиране при търсене на дестинации

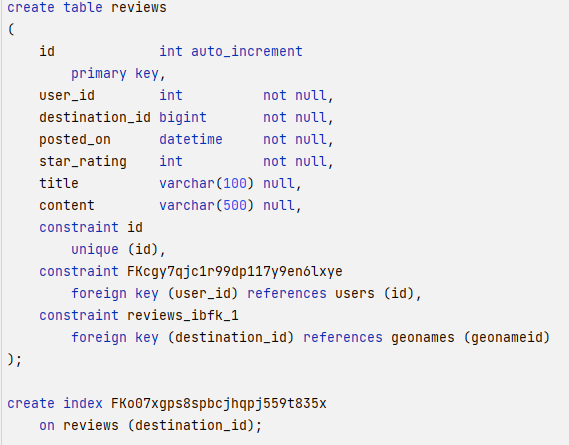
Фигура 2.4 – MySQL скрипт за създаване на таблицата geonames

* + - * 1. feature\_codes – съхранява FeatureCode
* code – еднобуквен код

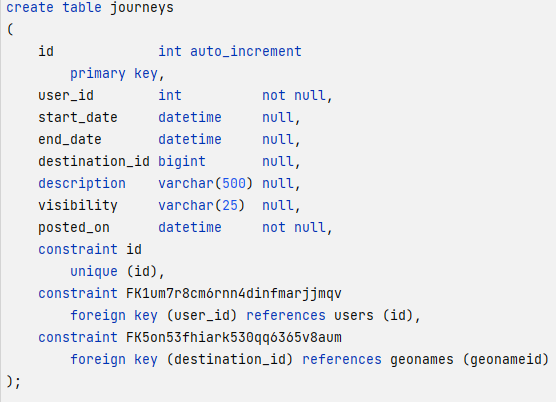
Фигура 2.6 – MySQL скрипт за създаване на таблицата feature\_codes

* meaning – значение на кода
* feature\_class – код, характеризиращ вида на дестинацията
* feature\_class\_meaning – значение на feature\_class колоната
* description – описание на feature\_class
  + - * 1. countries – съхранява CountryEntity
* country\_code – двубуквен код на държавата по стандарта ISO 3166. Това е първичния ключ на таблицата

Фигура 2.7 – MySQL скрипт за създаване на таблицата countries

* country\_name – име на държавата
* capital – столица
* area – площ
* continent – континент
  + - * 1. reviews – съхранява ReviewEntity
* id – първичен ключ на таблицата
* user\_id – външен ключ, сочещ към създателя на ревюто
* destination\_id – външен ключ, сочещ към дестинацията, за която е написано ревюто
* star\_rating – оценка от 1 до 5
* title – заглавие на ревюто
* content – съдържание

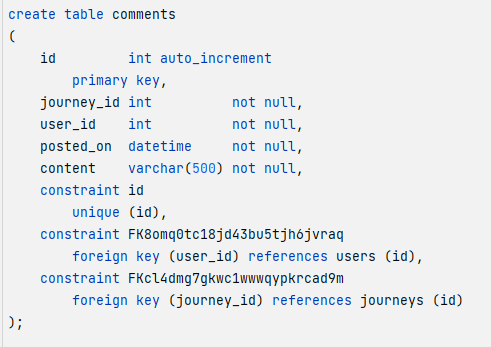
Фигура 2.8 – MySQL скрипт за създаване на таблицата reviews

* + - * 1. journeys – съхранява JourneyEntity
* id – първичен ключ на таблицата
* user\_id – външен ключ, сочещ към потребителя, собственик на това пътуване
* posted\_on – дата на публикуване
* start\_date – начална дата
* end\_date – крайна дата
* destination\_id – id на дестинацията, външен ключ
* description – описание
* visibility – поверителност на пътешествието

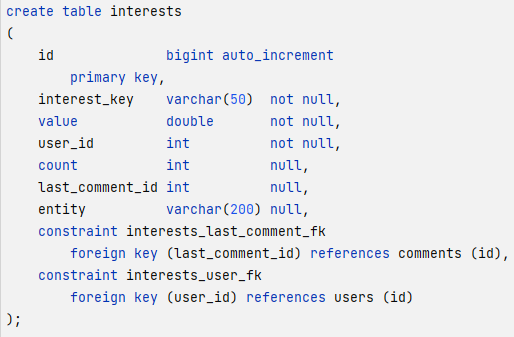
Фигура 2.9 – MySQL скрипт за създаване на таблицата journeys

* + - * 1. activities – съхранява ActivityEntity
  + id – първичен ключ
  + journey\_id – външен ключ към пътешествието, на което принадлежи
  + date – дата на извършване на дейността
  + type – вид дейност
  + description – описание
  + location – място
    - * 1. comments – съхранява CommentEntity

Фигура 2.10 – MySQL скрипт за създаване на таблицата activities

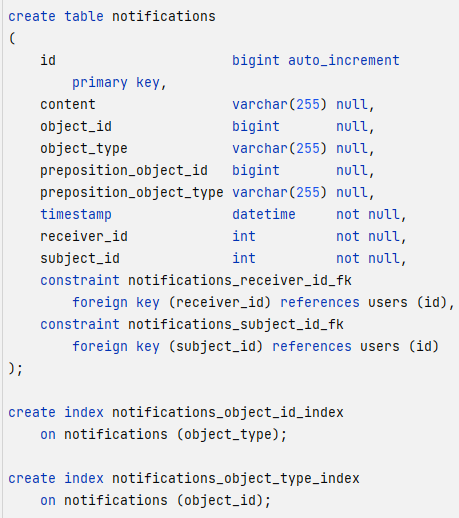
* + id – първичен ключ
  + journey\_id – външен ключ към пътешествието, което е коментирано
  + user\_id – външен ключ към потребителя, който е качил коментра
  + posted\_on – точна дата, на която е качен коментарът
  + content – съдържание на коментара

Фигура 2.11 – MySQL скрипт за създаване на таблицата activities

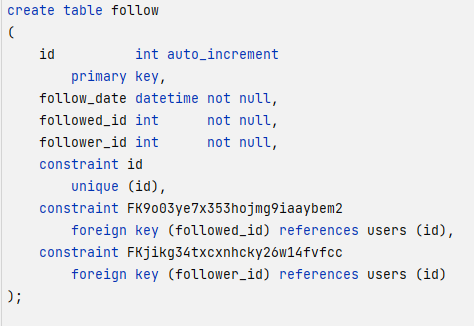
* + - * 1. interests – съхранява InterestEntity
* id – първичен ключ
* interest\_key – код на характеристиката, от която се интересува даден потребител
* value – средно-аритметично настроение, насочено от потребителя-притежател на интереса към характеристиката interest\_key
* user\_id – външен ключ към потребителя, за когото се отнася интереса
* count – брой на анализираните до този момент коментари
* last\_comment\_id – последно анализиран коментар. Заедно с count и value, играят роля в пресмятането на средно-аритметичният интерес към дадена характеристика на пътешествие
* entity – вид на обекта, към който се отнася този интерес

Фигура 2.12 – MySQL скрипт за създаване на таблицата interests

notifications – съхранява NotificationEntity

* id – първичен ключ
* content – съдържание на известието. Представлява изречение със едно сказуемо (просто изречение)
* object\_id – id на обектът, върху който е приложено действието
* object\_type – вид на обекта, върху който е приложено действието, напр. „comment” в изречението “alex posted a comment to your journey”. Играе ролята на допълнение в изречението content
* preposition\_object\_\_type – тип на обекта, за който се отнася object\_id, напр. „journey” в изречението “alex posted a comment to your journey”
* preposition\_object\_id – id на обекта, за който се отнася object\_id
* timestamp – отрязък от времето, в който събитието на известието се случва
* receiver\_id – външен ключ към получателя на известието
* subject\_id – външен ключ към предлога на изречението в content, който е извършителя на действието

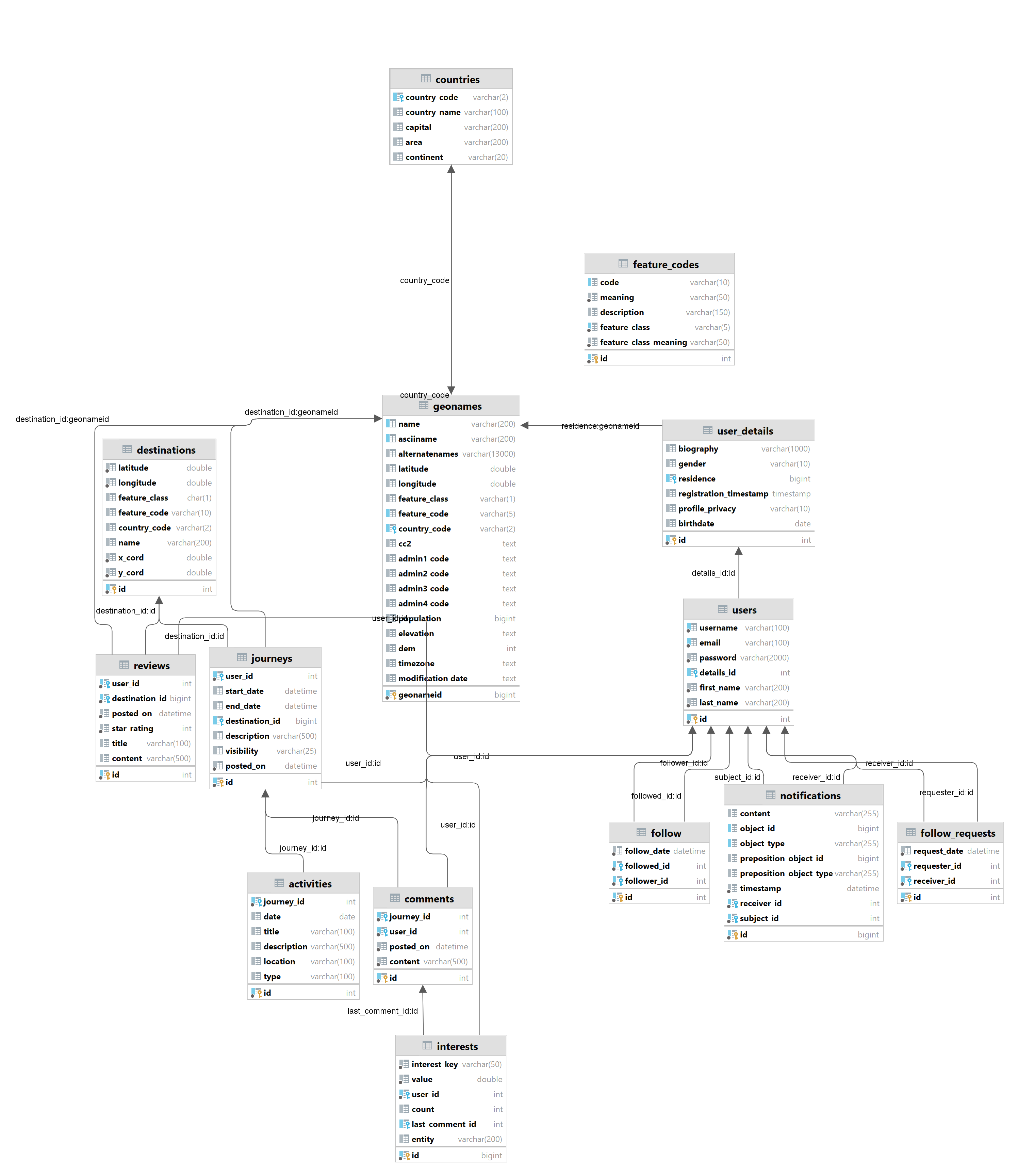
Фигура 2.13 – MySQL скрипт за създаване на таблицата notifications

* + - * 1. follows – съхранява FollowEnity
* id – първичен ключ
* follower\_id – външен ключ към последователя
* followed\_id – външен ключ към потребителя, който бива следван
* follow\_date – отрязък от времето, в който последователят е започнал да следва followed потребителя

Фигура 2.14 – MySQL скрипт за създаване на таблицата follows

* + - * 1. follow\_requests – съхранява FollowRequestEntity
* id – първичен ключ
* receiver\_id – външен ключ към потребителя, получател на заявката за следване

Фигура 2.14 – MySQL скрипт за създаване на таблицата follow\_requests

* requester\_id – външен ключ към потребителя, който заявява, че иска да последва receiver потребителя
* request\_date – отрязък от времето, в който е пусната заявката
  + - 1. Връзки между таблиците

Фигура 2.16

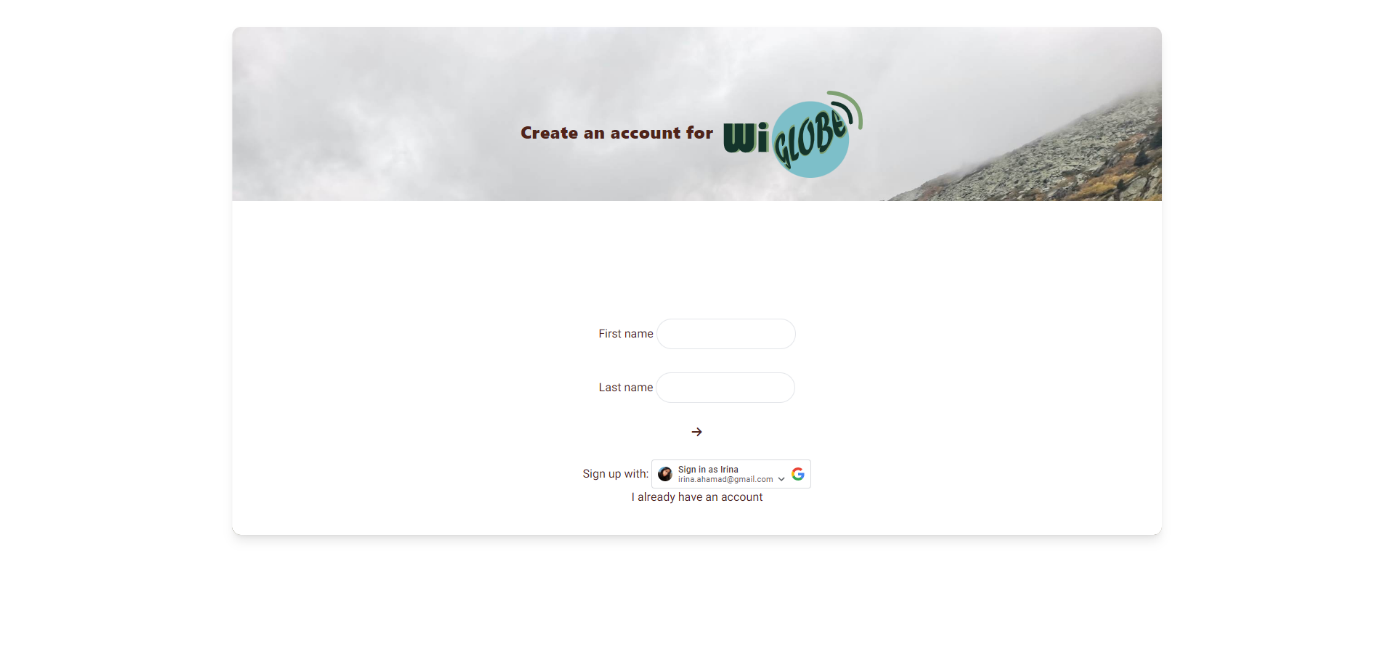
На фигура 2.16 е изобразена базата данни, под формата на таблици и техните връзки. Всяка стрелка представлява външен ключ, който свързва запис от таблицата от единия край на стрелката със запис от таблицата от другия край. Имената на стрелките са имена на колоните, които са външни ключове.

* + 1. Сценарий на уеб страницата
       1. Страница за аутентикация – „Log in“ страница

Фигура 2.17

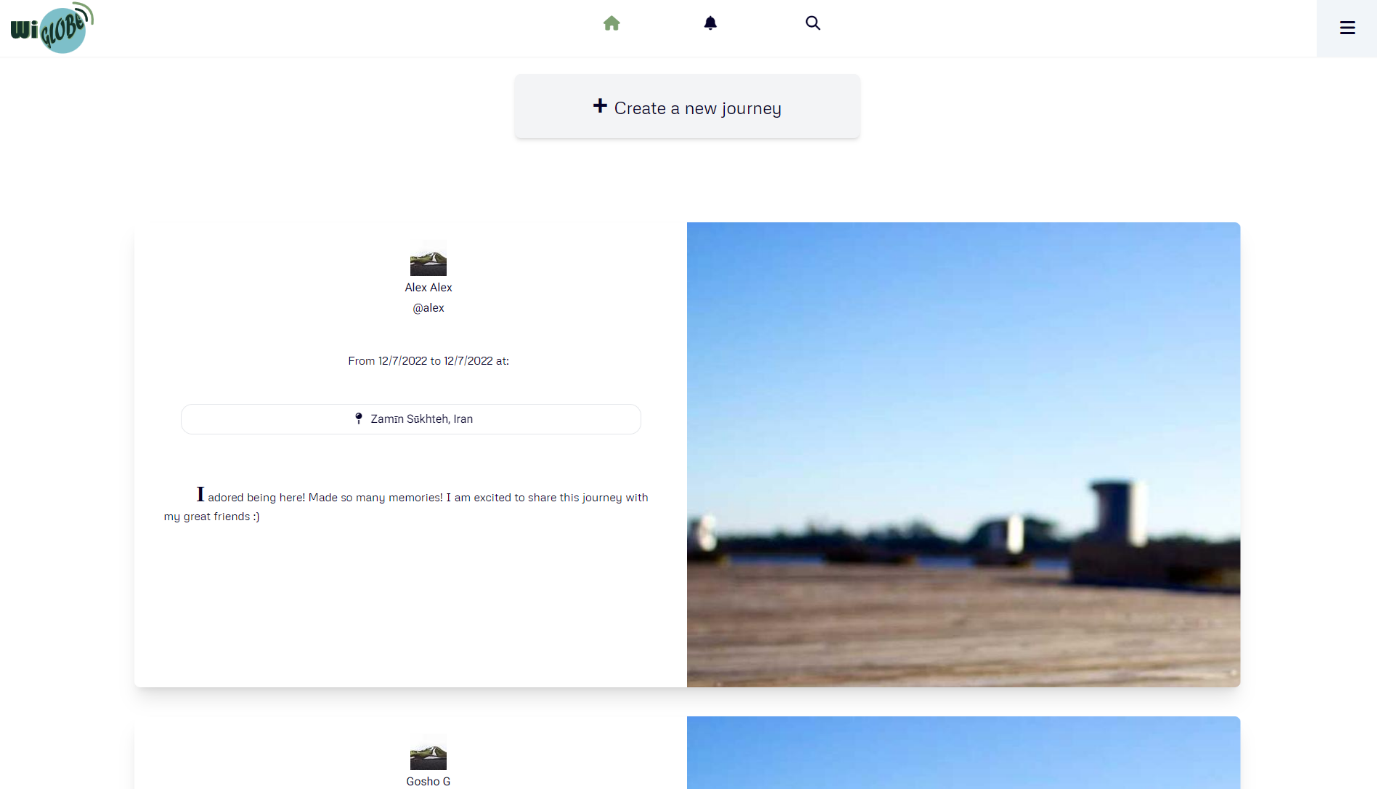
Пътят на таи страница е „/login”. Преди да достъпи съдържанието на приложението, човекът трябва да влезе в своя профил за да бъде потвърдена самоличността му в контекста на приложението. „Log in“ страницата е показана на фигура 2.17. За улеснението му, той има възможността да бъде аутентикиран чрез неговия Google акаунт. Ако няма профил, може да създаде такъв автоматично чрез Google бутона или да натисне върху “I don’t have an account”, което ще го препрати към страницата за регистрация.

* + - 1. Страница за регистрация

Страницата за регистрация се намира на “/signup” (фигура 2.18). Тя дава възможност за връщане към „Log in“ страницата, регистрация чрез Google или регистрация чрез въвеждане на данни.

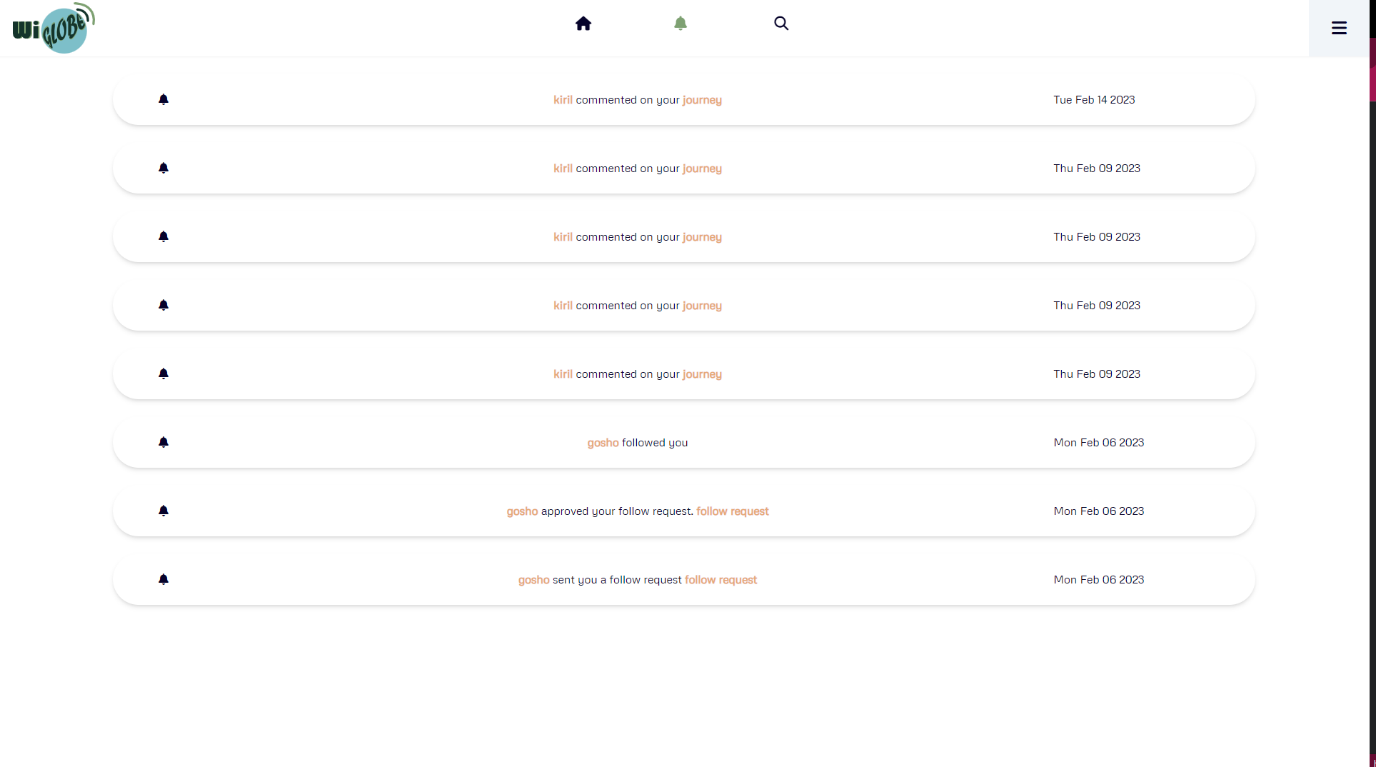
Фигура 2.18

* + - 1. Начална страница

Началната страница представлява стена с препоръчани за потребителя пътешествия (фигура 2.19)

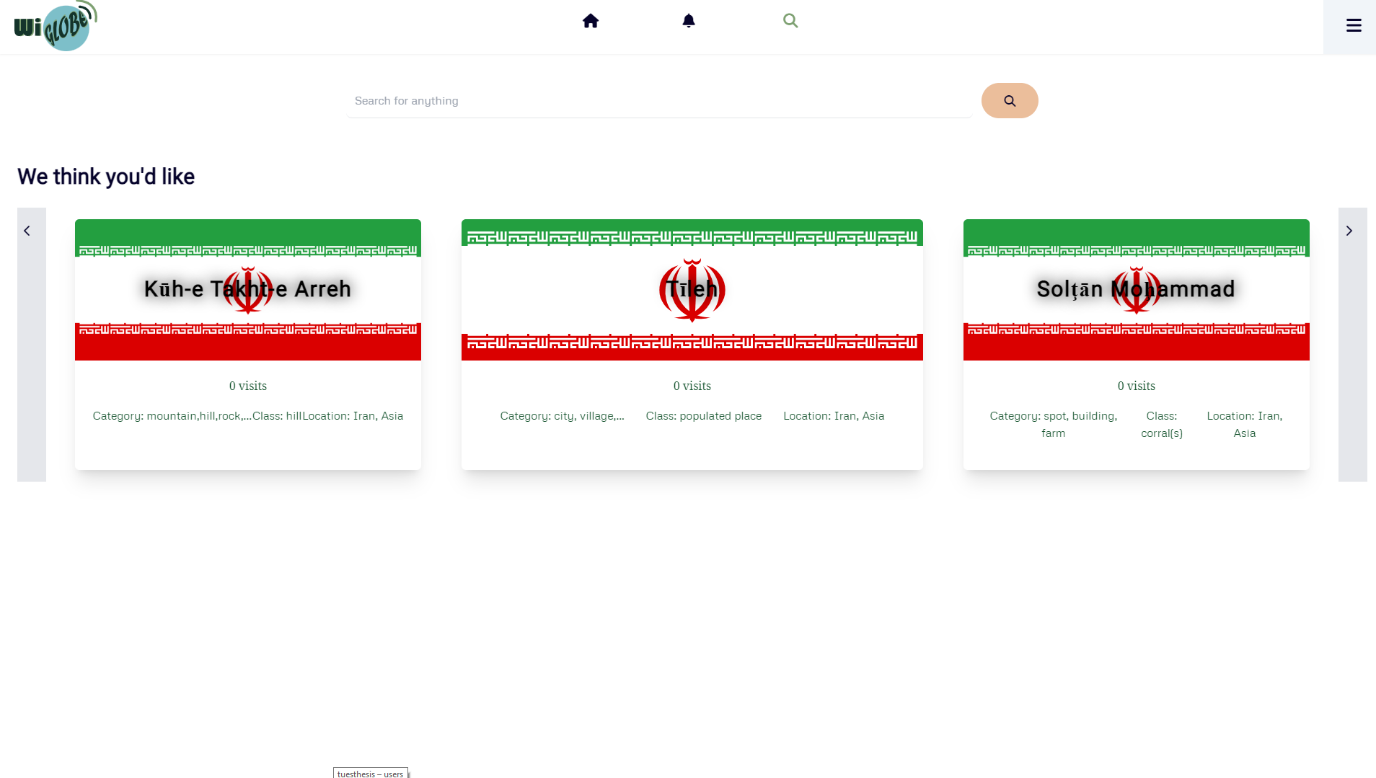
Фигура 2.19

* + - 1. Известия

До страницата със известия („/notifications”) се стига чрез клик върху камбанката в навигационния елемент.

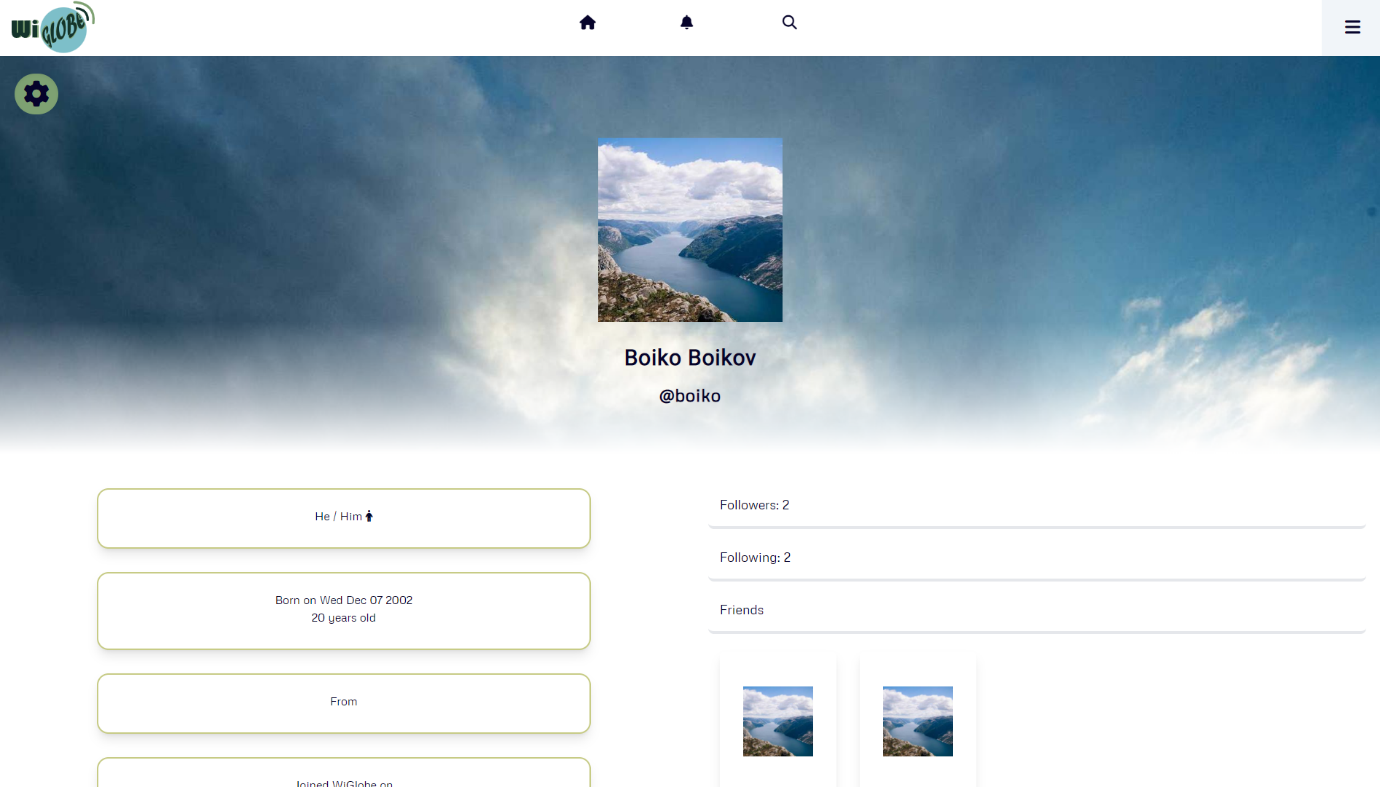
Фигура 2.20 – Страница за нотификации

* + - 1. Explore страница – страница за търсене

Тази страница позволява свързването на потребителя с други хора. Пътят към страницата е: „/explore” и до нея може да се достигне чрез навигационния елемент.

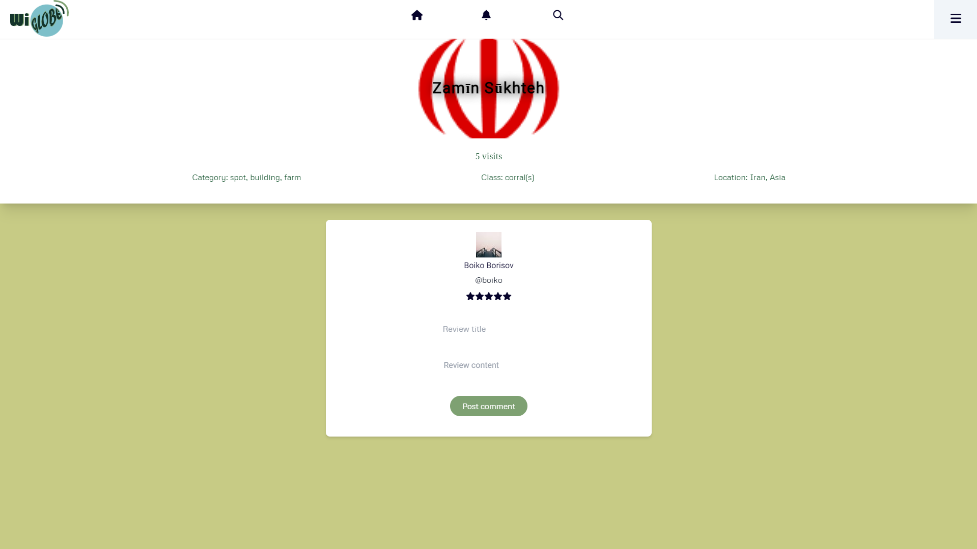
Фигура 2.21 – страницата explore

* + - 1. Профил

Страницата за профил дава информация за един потребител – за самоличността му (в рамките на информацията, която се е съгласил да предостави), за взаимоотношенията му с други потребители и неговите пътешествия.

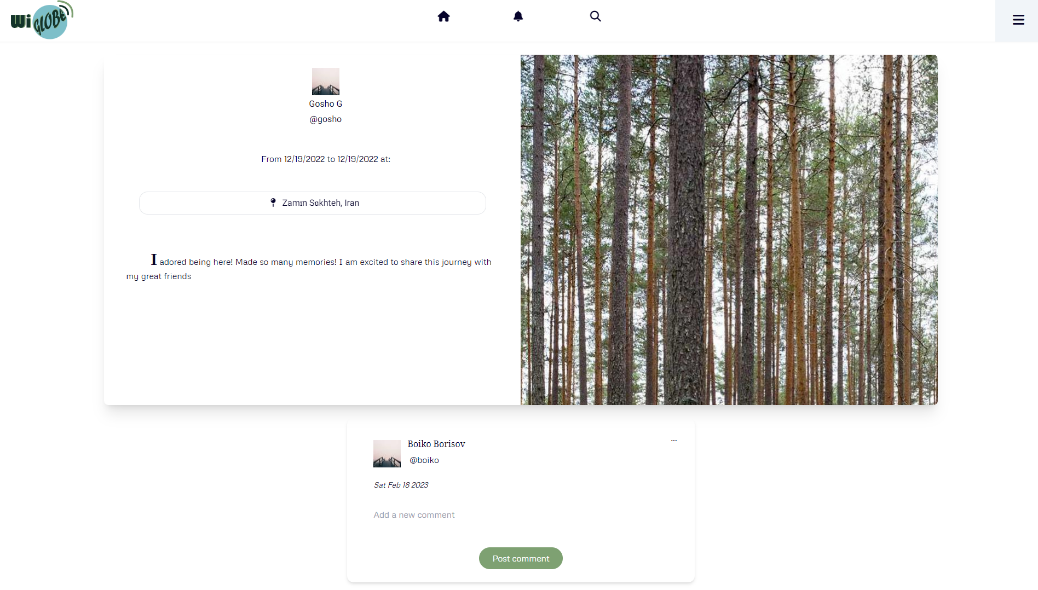
Фигура 2.23 – Профилна страница

* + - 1. Страница на дестинация

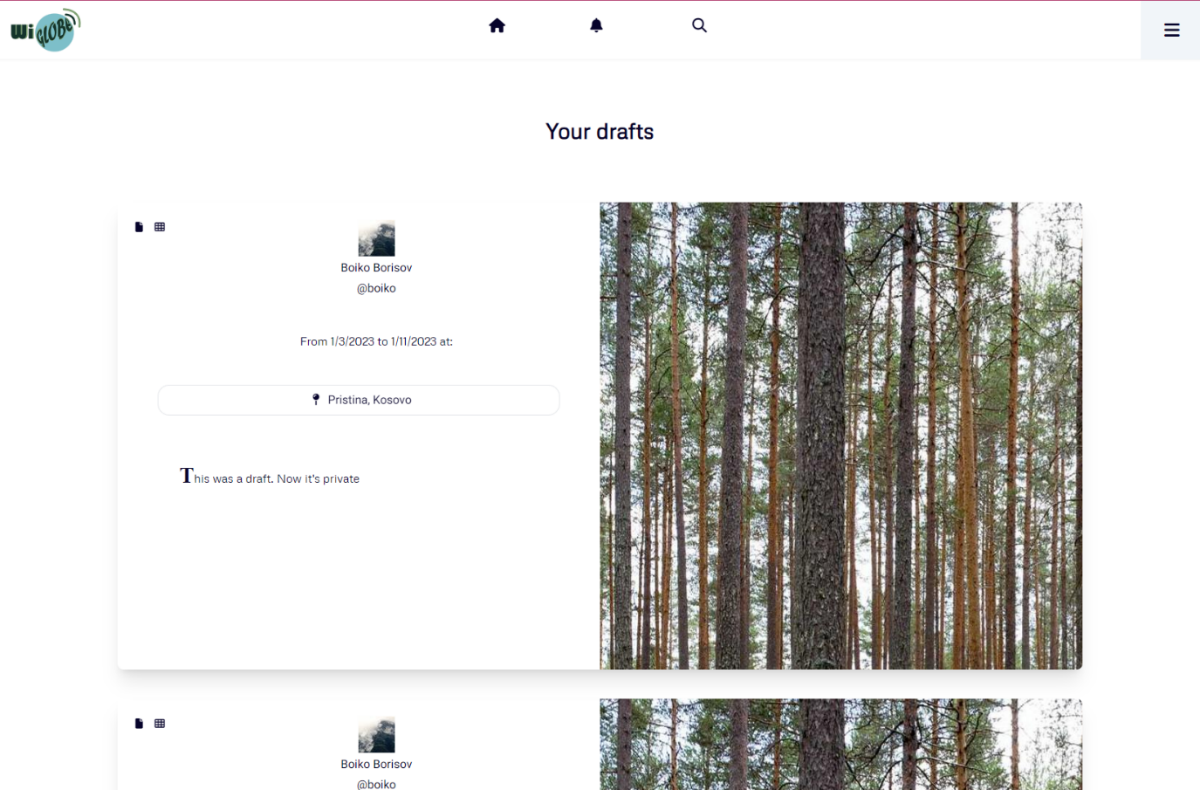
На страницата за дестинация, намираща се на „destinations/{destination\_id}”, един потребител може да получи повече информация за дестинация, например за местоположението ѝ или за средната ѝ оценка, дадена от други потребители. Освен това, може да остави ревю със своето мнение.

Фигура 2.24 – Страница на дестинация

* + - 1. Страница на пътешествие

Страницата за пътешествие, на пътя „journeys/{journey\_id}” един потребител може детайлно да разгледа едно пътешествие и да остави коментар.

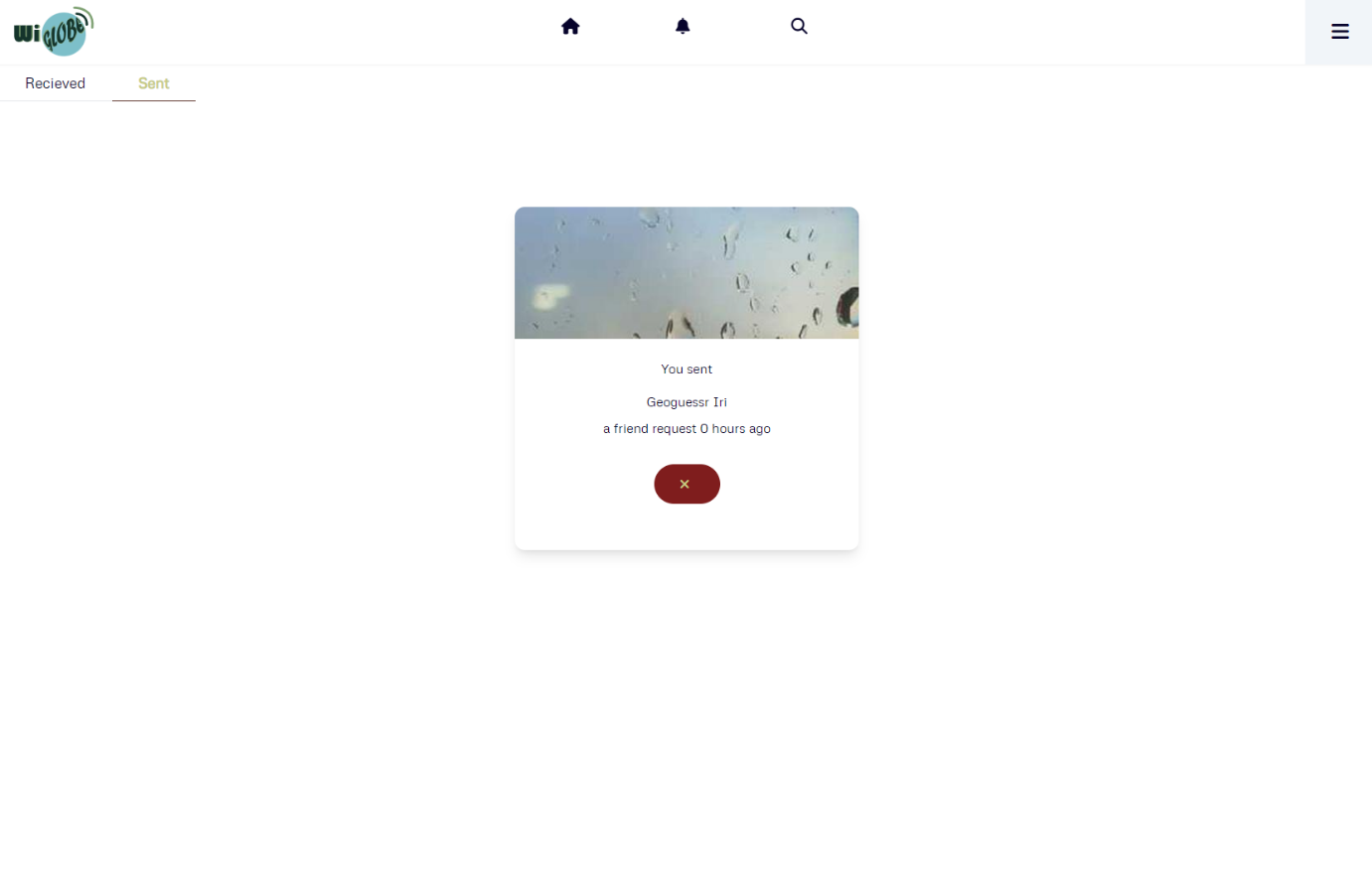
Фигура 2.25 – Страница на пътешествие

* + - 1. Страница за непубликувани пътешествия

Фигура 2.26 – Страница за чернови – Drafts страницата

Непубликуваните пътешествия са чернови, които се запазват и позволяват на потребителя да ги публикува.

* + - 1. Страница с заявки за следване

На страницата със заявки за следване потребителят може да разгледа или прекрати изходящите си заявки за следване или да потвърди или откаже входящите такива.

Фигура 2.27 – Страница със заявки за следване

ТРЕТА ГЛАВА

Програмна реализация

* 1. Реализация на приложно-програмния интерфейс
     1. Entities

В контекста на Spring JPA entityt е обект, който е съпоставен с таблица от базата данни и капсулира данните и бизнес логиката на съответната таблица. Анотацията @Entity се използва за маркиране на класа като entity, коeто може да бъде съхраненo в таблица на база данни.

Aнотацията @Table се използва за посочване на таблицата от базата данни, която съответства на клас същност. Всеки параметър от класа се асоциира с колона от съответната таблица чрез @Column.

@ManyToOne и @OneToMany са анотации, използвани в JPA за дефиниране на връзки между entity класове, като @ManyToOne показва връзка много към едно от един обект към друг, а @OneToMany показва връзка един към много от един обект към множество на друг обект. Анотацията @JoinColumn се използва за посочване на колоната с външен ключ в таблицата на обектът, който притежава връзката.

Пример е дефинираната връзка @ManyToOne между JourneyEntity и UserEntity, където всяко Journey може да има само един потребител-създател, но един потребител може да има много JourneyEntity-та. В този случай JourneyEntity има анотация @ManyToOne, отнасяща се до UserEntity, а същността UserEntity ще има анотация @OneToMany, отнасяща се до JourneyEntity. Анотацията @JoinColumn се използва за определяне на колоната на външния ключ в таблицата за JourneyEntity-та - journeys, която препраща към таблицата users.

* + - 1. ActivityEntity

Този клас представлява дейност, извършвана по време на пътешествие и по тази причина е асоцииран с класа JourneyEntity с връзка many to one – много към едно (фигура 3.1). С други думи, едно JourneyEntity може да има много ActivityEntitties. Колоната с външен ключ към таблицата journeys, сочеща към JourneyEntity-то, което е „родител“ на ActivityEntity-to, е non-nullable, което означава, че стойностите в колоната journey\_id или стойноститте на параметъра journey, не могат да бъдат празни. Това е защото ActivityEntity не може да съществува без JourneyEntity като родителски клас.

Фигура 3.1

**За улеснение и по-чист и четим код, е създаден конструктор, който приема като параметър ActivityRequest и JourneyEntity. Констуктора създава ActivityEntity и е показан на фигура 3.2.

Фигура 3.2

* + - 1. CommentEntity

Кодът на фигура 3.3 дефинира клас CommentEntity, който представлява коментар в система за коментиране. Той съдържа полета за id, journey, user, postedOn и content. Класът има и вторичен конструктор (фигура 3.4), който приема CommentRequest, JourneyEntity и UserEntity като аргументи, за да създаде нов обект CommentEntity със зададената информация.

Фигура 3.3



Фигура 3.4

* + - 1. CountryEntity

На фигура 3.5 е дефиниран клас CountryEntity с полета, представляващи свойствата на дадена държава. Полетата countryCode и countryName са маркирани като nullable = false, което означава, че те не могат да бъдат празни. Другите полета capital, area и continent са nullable, което означава, че те могат да бъдат празни.

Фигура 3.5

* + - 1. DestinationEntity

Фигура 3.6

DestinationEntity, представлява ред от таблица с име "geonames" в базата данни. Тя има различни параметри като id, latitude, longitude, name, asciiname, alternatenames, feature\_class и feature\_code. Класът също така има връзка ManyToOne с CountryEntity, както и връзка OneToMany със същности, наречени ReviewEntity и JourneyEntity. Кодът за DestinationEntity е на фигура 3.6.

* + - 1. FeatureCode
      2. FollowEntity
      3. FollowRequestEntity
      4. InterestEntity
      5. JourneyEntity
      6. NotificationEntity
      7. ReviewEntity
      8. UserDetailsEntity
      9. UserEntity



* + 1. Контролери
       1. ActivityController
       2. CommentController
       3. DestinationController
       4. FollowController
       5. FollowRequestController
       6. JourneyController
       7. NotificationController
       8. ReviewController
       9. UserController
    2. Repositories
       1. ActivityRepository
       2. CommentRepository
       3. CountryRepository
       4. FeatureCodesRepository
       5. FollowRepository
       6. FollowRequestRepository
       7. InterestsRepository
       8. JourneyRepository
       9. NotificationRepository
       10. ReviewRepository
       11. UserDetailsRepository
       12. UserRepository
    3. Модели
       1. Модели за входящи заявки
       2. Модели за отговори на заявки
       3. Проекции

В контекста на Spring Data JPA проекцията се отнася за механизъм, който позволява избора на подмножество от атрибути на дадена структура, което позволява извличането само на необходимите данни, а не на цялата структура. Проекциите се създават като интерфейси, които дефинират желаното подмножество от полета и са полезни за оптимизиране на производителността на заявките и минимизиране на количеството данни, прехвърляни между базата данни и приложението.

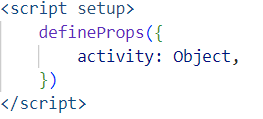
* + 1. Пакетът „security“
    2. Пакетът „recommender”
    3. Тестове
       1. Тестове на контролери
       2. Тестове на сървиси
  1. Създаване на уеб приложение
     1. Компоненти

Компонентите във Vue 3 и Nuxt 3 са модулни блокове за многократна употреба, които капсулират функционалността и представянето на елементите на потребителския интерфейс. Компонентите могат лесно да бъдат съставяни заедно, за да образуват по-големи и по-сложни интерфейси, и могат да бъдат персонализирани и разширявани чрез използване на предварително подадени променливи или слотове. Компонентите се намират в папката „componenets”.

Стилът на компонентите е приложен чрез Tailwind CSS класове. Често използвани в проекта предефинирани класове са:

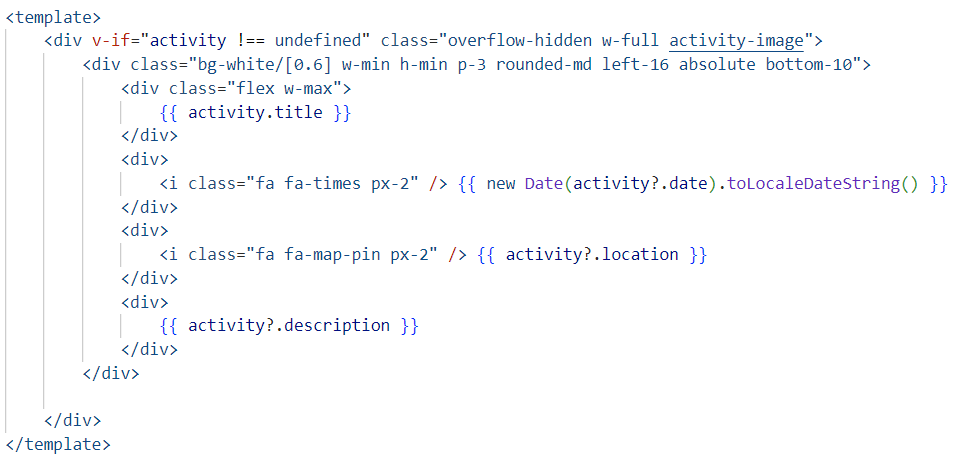
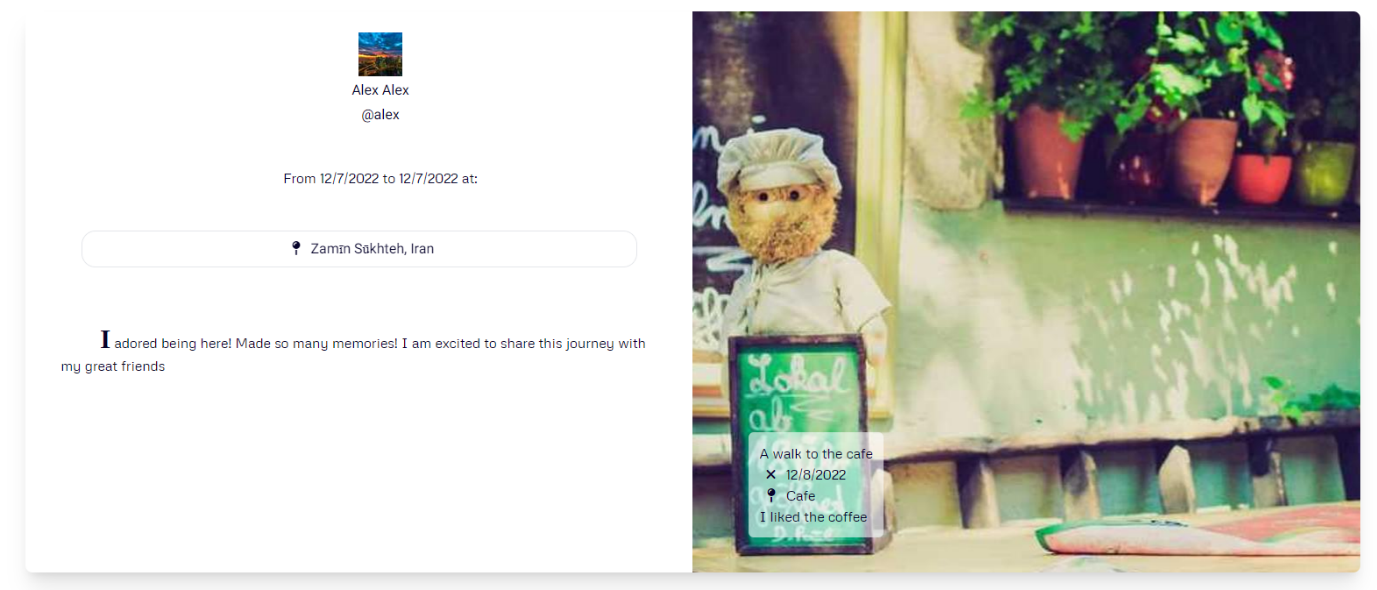
* „w-full“ – елемента заема цялата ширина на родителя си
* „flex“ – размера на елемент с този клас е гъвкав, а не фиксиран
* „p-[число]“ – слага поле в елемента
* „m-[число]“ – отстояние между съседните елементи
* „rounded“ – заобля върховете на елемента
* „shadow“ – слага сянка около елемента
* „relative“ – позиция, зависеща от съседни елементи
* „absolute“ – позиция, независеща от съседни елементи
* „bg-[цвят]“ – прилага цвят на фона
* „overflow-hidden“ – скрива преливащата част на елементите, излизащи извън родителя си
* „hover:[стил]“ – докато курсора е върху елемента, прилага стила
* „focus:[стил]“ – при натискане върху елемента, прилага стила
* „text-lg“ – уголемява текста
* „font-bold“ – удебелява шрифта

Някои от компонентите ползват икони от сайтът Font awesome. Начинът, по който работи Font awesome CDN-ът е показан на фигура 3. В тагът за икона <i> е поставен клас „fa”, който означава, че ще бъде използван Font Awesome клас, и „fa-times”, който се грижи за илюстрацията на самата икона – в този случай часовник.

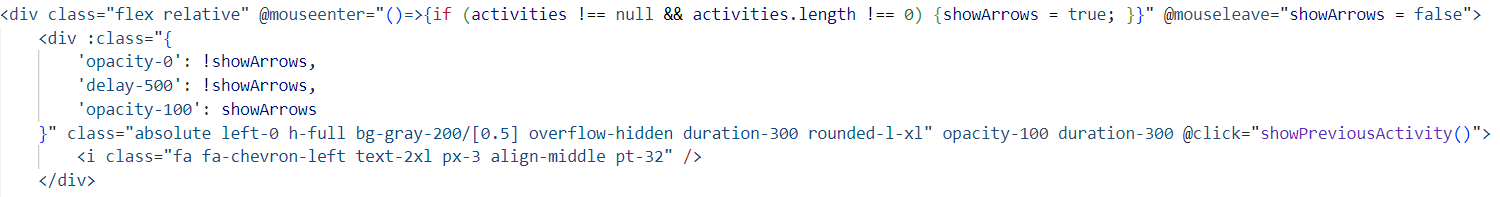
* + - 1. Activity

Activity компонентът служи за илюстриране на дейност (ActivityEntity). Получава prop-ове, за да покаже определено ActivityEntity, по начина, показан на фигура 3. На фигура 3., компонентът получава информацията, че prop-ът, който трябва да очаква е от тип Object – той трябва да бъде обект. Prop-овете се използват в <template> тага. Пример за <template> таг е показан на фигура 3, където се създава компонентът чрез HTML.

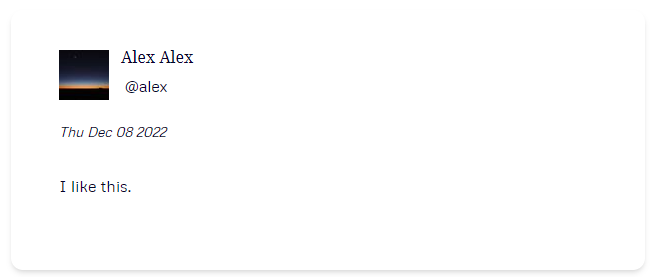
Фигура **Error! No text of specified style in document.**.2

В този конкретен <template> е вложен <div> таг, който служи като контейнер и преди да бъде рендериран, проверява дали стойността на prop-а activity е undefined. В скучай, че е undefined, то <div> тагът няма да бъде рендериран, а ако activity има стойност, то <div> тагът и неговото съдържание ще бъдат показани на клиентската уеб страница. Стойностите на prop-овете се четат от <template> чрез двойни фигурни скоби. В случая на Activity компонента, заглавието се достъпва чрез „{{ activity?.title }}”, датата – чрез „{{ activity?.date }} и т.н. За да бъдат предотвратени грешки, параметрите на activity prop-а се достъпват, като след „activity” и преди точката, се слага въпросителен знак. Въпросителният знак проверява дали activity е null или undefined, преди да бъде взет параметъра.

* + - 1. ActivityAdder

Функцията му е да позволи на потребителя да въведе информация, за дейност, извършвана по време на пътешествието си и да покаже въведените досега (в това пътешествие) дейности. Когато дейността е завършена и потребителя кликне върху „Add activity”, компонентът праща данните на компонента JourneyAdder (точка 3.2.1.10). За да се прегледат прикачените към пътешествието Activity-та, има стрелки, като към всяка има функция, която слага на показ предишно или следващо Activity. Стрелките са скрити, докато курсора не е върху компонента, за по-чист вид. Начинът, по който е постигнат този механизъм е на фигура.

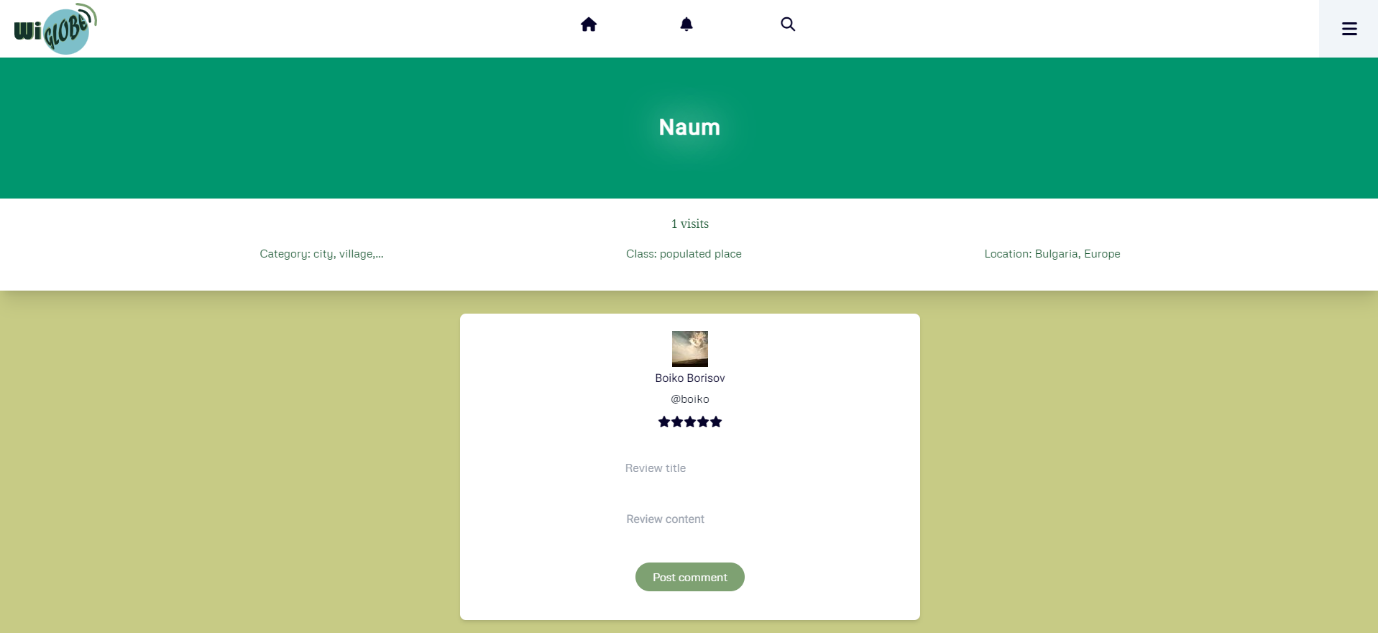
* + - 1. Comment

Компонентът за коментар показва коментар, подаден му чрез prop. Ако коментарът принадлежи на текущия потребител, човекът има възможността да изтрие коментара чрез клик върху трите точки и след това върху “Delete”.

* + - 1. CommentAdder

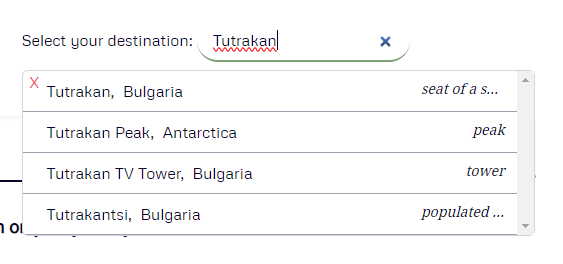
CommentAdder компонентът предоставя интерфейс, през който потребител може да създаде коментар и да го качи под пътешествие. За да се покаже името на качващия коментар, се прави инстанция на UserStore, където се пазят данните за текущия потребител. Когато се кликне „Post comment”, се изпълнява функцията emitPost (фигура 3), която от своя страна задейства функция от родителя на компонента, пращаща POST заявка към сървъра и коментарът бива качен.

* + - 1. Destination

Destination показва дестинация с подадени от родителския компонент prop‑ове. На фигура 3 е показан компонентът на страницата „destinations/[destinaionId]” заедно със компонентът ReviewAdder (отдолу).

* + - 1. DestinationMini

DestinationMini e малка версия на Destination, чиято цел е по прост и стилен начин да даде кратка информация за крайната точка на пътешествие.

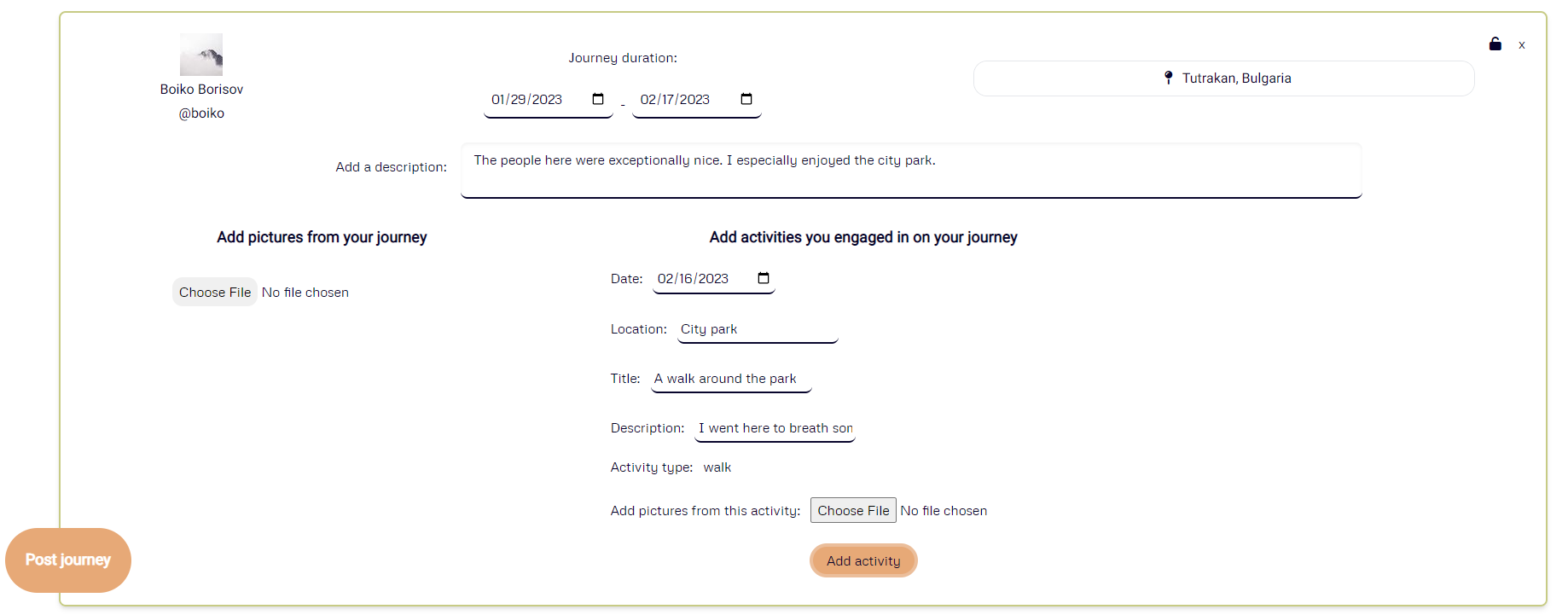
* + - 1. DestinationSearchResult

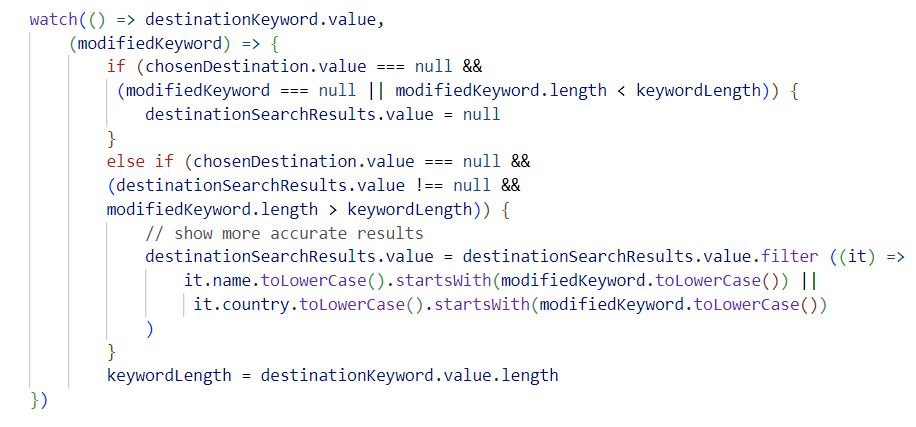
DestinationSearchResult се използва, за да визуализира резултатите от търсене на дестинация по време на създаването на пътешествие. След клик върху избраната дестинация, тя се селектира като дестинация на създаваното пътешествие. Ако потребителя иска да затвори резултатите от търсене без да е селектирал дестинация, може да кликне върху червения хикс.

* + - 1. FollowRequest

Този компонент може да бъде срещнат във follow requests страницата.

* + - 1. Journey
      2. JourneyAdder

JourneyAdder компонентът е интерфейс за създаване на journeys – пътешествия. Избират се начална и крайна дата, дестинация, поставя се описание и

За да се оптимизира приложението, се намалява броя на заявки за търсене на дестинации. Това се случва като след получен резултат от търсене, към думата, с която се търси, се въведат допълнителни символи. Алгоритъмът на фигура филтрира получените резултати и DestinationSearchResult ги изобразява.

* + - 1. JourneyAttachments

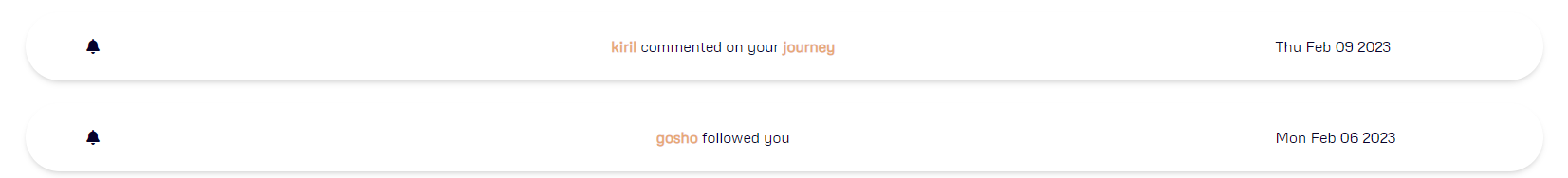
Показва снимките и дейностите, свързани с едно пътешествие.

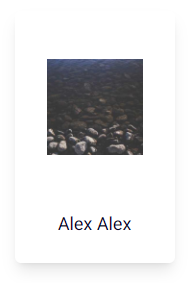
* + - 1. JourneyMenu

Дава възможност на потребителя да обработва или трие своите пътешествия.

* + - 1. Navigation

В горния край на страницата, спомага лесната навигация между страниците.

* + - 1. Notification

Този компонент получава известие като prop и го визуализира, като на фигура 3 е показан начинът, по който се избира къде да се постави линк. Пример за избираемото поставяне на линкове е на фигура 3. Думата „kiril” е връзка към профила на потребител с потребителско име „kiril”. Думата „journey” е връзка към страницата на съответното пътешествие. „gosho” е връзка към профила на „gosho”.

* + - 1. ProfileMini

Този компонент улеснява показването на първостепенни данни за един потребител – имената, потребителското му име, профилната му снимка – фигура 3.

* + - 1. Review

Review получава ревю под формата нa prop и го визуализира.

* + - 1. ReviewAdder

Компонент за публикуване на ревю – поставя се заглавие, съдържание на ревюто и оценка в звезди от 1 до 5. При клик върху Post бутона, ревюто бива качвано чрез POST заявка.

* + 1. Страници
       1. login.vue
       2. signup.vue
       3. index.vue
       4. profile
          1. profile/me
          2. profile/[username]
       5. follow-requests.vue
       6. notifications.vue
       7. explore.vue
    2. Папката layouts

В Nuxt layout-ите или с други думи, шаблони, са функционалност, която позволява на разработчиците да определят общата структура и оформление на страниците на своето приложение. Разположенията могат да се използват за дефиниране на заглавието, долния колонтитул и други общи елементи на дадена страница, както и за дефиниране на цялостната структура и поведение на страницата. Със системата за оформление на Nuxt многократно използваеми и персонализируеми оформления на страници се реализират по лесен и семпъл начин.

На фигура 3 е показан шаблонът, който повечето страници използват. Той се грижи store-овете да са подготвени и да показва Navigation компонента в горната част на страницата.

* + 1. Middleware

В контекста на Nuxt middleware (междинен софтуер) е функция, която позволява на разработчиците да изпълняват функции при всяка заявка за страница, преди тя да бъде визуализирана. Middleware-ът на Nuxt може да се използва за изпълнение на задачи като удостоверяване, извличане на данни от страна на сървъра или промяна на контекста на страницата и може да се дефинира глобално за всички страници или локално за конкретни страници или маршрути.

 На фигура 3, middleware-ът проверява дали има валиден потребител, влязъл в профила. Ако има, не е нужно да минава през login страницата и може да продължи направо към home страницата.

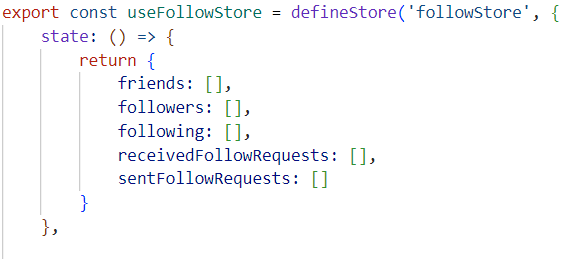
* + 1. Папката stores

За запазването на state-a на приложението е използвана Pinia.

* + - 1. UserStore

UserStore-a пази данни за логнатия потребител и се грижи за Log In, Sign up и Log out функционалностите. В state са променливи, които се пазят между обновяванията на страницата, а в actions са методите (фигура). Показана е signUp функцията, която използва функция, пращаща заявка за регистрация към сървъра. След като изчака и получи отговор, той се запазва в константната променлива response.

* + - 1. FollowStore

Целта на FollowStore е да пази информация за обвързването между логнатия потребител и другите потребители, за да се осигури бързодействие, когато има нужда от проверки. Например, ако даден профил е с PRIVATE поверителност или с други думи заключен за неследващи го хора, когато потребителя отиде на профилната му страница, не трябва да прави допълнително запитване към сървъра за поверителността на профила. Ако се знае (от FollowStore) дали логнатия потребител следва профила или дали е изпратена заявка за следване към този профил, може директно да се заредят нужните данни. На фигура 3 са показани запазената информация: приятели, последователи, последвани потребители, получени и изпратени заявки за следване.

* + - 1. JourneyStore

JourneyStore пази пътешествията на логнатия потребител и тези, които се показват на стената му на началната страница. По този начин се оптимизира приложението – home страницата е типично често достъпвана страница.

* + 1. Папката js

В папката js са файлове с функции, които използват axios, за да правят заявки към сървъра.

Методите на axios, използвани в приложението са следните:

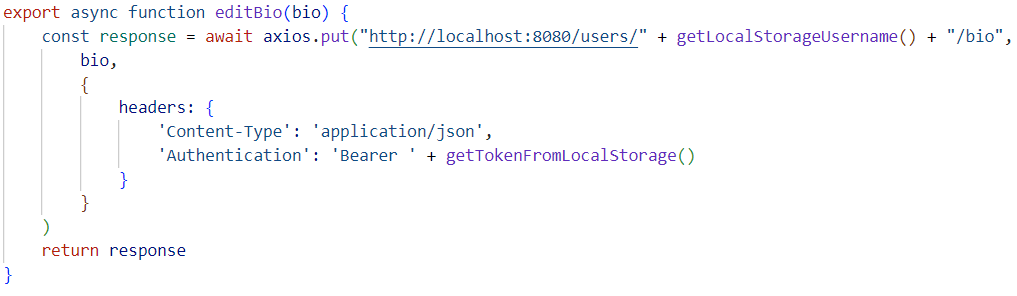
* post(url, body, config):

Връща отговора на заявката под формата на обект. Пример за POST заявка, направена с axios.post() е показана на фигура 3, където се изпращат данни за потребителя, за да може да влезе в профила си.

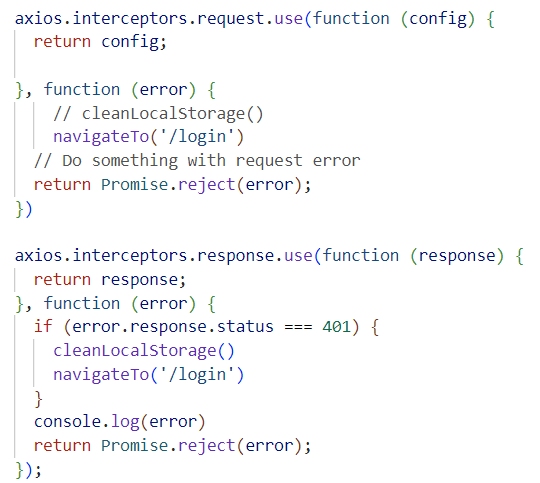
* get(url, config)

Методът за GET заявка на axios връща отговора на заявката под формата на обект. Демонстриран е на фигура 3, където се взимат детайлите на потребител от сървъра

* put(url, body, config)

С този метод се правят PUT заявки. Функцията на фигура 3 показва как се променя биографията на потребител. Връща отговора на заявката под формата на обект.

* delete(url, config)

На фигура 3 се прави DELETE заявка, за да се изтрие профила на потребителя. Връща отговора на заявката под формата на обект.

* Interceptors

Interceptor-ите обработват заявки преди или след изпращането им. Например, на фигура 3 interceptor-а за отговор от заявка проверява дали статусът на заявката гласи, че потребителя не е влязъл в профила си с валидни данни. Тогава, го праща на login страницата, за да влезе в профила си.

В тази папка е помощния файл localStorageUtil.js, чрез който лесно се достъпват данните, съхранявани локално. На фигура 3 са съответно:

* Функцията за взимане на потребителското име getLocalStorageUsername()
* Функцията за взимане на токенът getTokenFromLocalStorage()
* Функцията cleanLocalStorage(), която трие данните, съхранени от приложението. Използва се при изход от профил.

Тези функции проверяват дали прозорецът е готов, за да не предизвикат грешка, когато се извиква localStorage.

ЧЕТВЪРТА ГЛАВА

Ръководство на потребителя

Заключение

Уеб платформата за споделяне на пътешествия, разработена за този проект, представлява съвременно решение за любителите-туристи, като интегрира съвременни технологии с удобен за потребителя интерфейс и интуитивен дизайн. Изградена с помощта на Spring Boot с Kotlin за бекенд, Nuxt 3 за фронтенд и MySQL за база данни, архитектурата и функциите на платформата са оптимизирани за производителност, всеобхватност и ангажираност на потребителите.

Интерфейсът на фронтенда е проектиран с фокус върху потребителското изживяване, като се използва Tailwind CSS рамката за рационализиране на процеса на стилизиране и създаване на адаптивни оформления, които се адаптират към различни размери на устройствата. Удостоверяването на потребителите е осигурено с помощта на Spring Security, което гарантира сигурен достъп до потребителските акаунти и осигурява допълнително ниво на защита срещу неоторизиран достъп.

Приложно-програмният интерфейс на платформата използва Spring Boot и Kotlin, за да създаде комуникационен слой между потребителите. Той управлява потока от данни между различните части на приложението и осигурява допълнителна функционалност, като анализ на настроенията. Използването на библиотеката VADER помага на платформата да осъществи обратната връзка с потребителите, което позволява на платформата да реагира по-добре на коментарите на потребителите.

Този проект демонстрира потенциала на съвременните технологии за революционизиране на транспортните услуги. Проектът постигна поставените цели и задачи чрез използването на различни технологии и принципи на програмиране в реални условия, като оставя възможности за по-нататъшно развитие и подобрения, за да се подобри неговата функционалност, ефективност и интерактивност.

Използвана литература

geonames. http://www.geonames.org/.

Съдържание

[**TЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ** 1](#_Toc127517864)

[Използвани термини, съкращения и абревиатури 5](#_Toc127517865)

[ТРЕТА ГЛАВА 36](#_Toc127517866)

[3.1 Реализация на приложно-програмния интерфейс 36](#_Toc127517867)

[3.1.1 Entities 36](#_Toc127517868)

[3.1.2 Контролери 36](#_Toc127517869)

[3.1.3 Repositories 36](#_Toc127517870)

[3.1.4 Модели 36](#_Toc127517871)

[3.1.5 Пакетът „security“ 36](#_Toc127517872)

[3.1.6 Пакетът „recommender” 36](#_Toc127517873)

[3.2 Създаване на уеб приложение 36](#_Toc127517874)

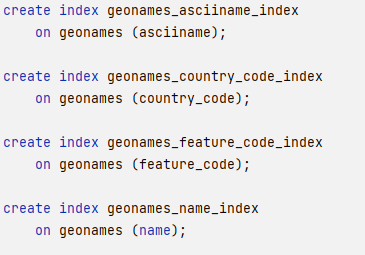
[ЧЕТВЪРТА ГЛАВА 37](#_Toc127517875)

[Заключение 38](#_Toc127517876)

[Използвана литература 39](#_Toc127517877)

[geonames. http://www.geonames.org/. 39](#_Toc127517878)

[Съдържание 39](#_Toc127517879)

1.  [↑](#endnote-ref-1)
2. [↑](#endnote-ref-2)