ЧАСТНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ЗА ДИГИТАЛНИ НАУКИ „СОФТУНИ БУДИТЕЛ“, гр. София

**ДИПЛОМЕН ПРОЕКТ**

на Атанас Димитров Вълчев

ученик/ученичка от XII клас

професия- код: **481030**, **“Приложен програмист”**

специалност- код: **4810301**, **“Приложно програмиране”**

**Тема: Разработване на уебсайт за предлагане и наемане на места за настаняване**

**Ръководител-консултант: Кирил Поповски**

Сесия: май-юни 2024г.

Дата:.........................

**СЪДЪРЖАНИЕ**

[1. Увод 4](#_Toc162734595)

[1.1. Описание 4](#_Toc162734596)

[1.2. Мотивация 4](#_Toc162734597)

[1.3. Основна цел 5](#_Toc162734598)

[1.4. Очаквания 5](#_Toc162734599)

[1.5. Предизвикателства 5](#_Toc162734600)

[2. Изложение 7](#_Toc162734601)

[2.1. Описание на софтуерните технологии 7](#_Toc162734602)

[2.1.1. Уеб приложение 7](#_Toc162734603)

[2.1.2. HTML 8](#_Toc162734604)

[2.1.3. CSS 9](#_Toc162734605)

[2.1.4. JavaScript 10](#_Toc162734606)

[2.1.5. React 12](#_Toc162734607)

[2.1.6. React Components 13](#_Toc162734608)

[2.1.7. JSX (JavaScript XML) 14](#_Toc162734609)

[2.1.8. Virtual DOM 15](#_Toc162734610)

[2.1.9. React Hooks 16](#_Toc162734611)

[2.1.10. React Router 17](#_Toc162734612)

[2.1.11. Material UI 18](#_Toc162734613)

[2.1.12. Framer Motion 19](#_Toc162734614)

[2.1.13. Мениджър на пакетите 20](#_Toc162734615)

[2.1.14. Firebase 21](#_Toc162734616)

[2.1.15. Използвани пакети 23](#_Toc162734617)

[2.2. Реализация на приложението 25](#_Toc162734618)

[2.3. Поставени цели за проекта 25](#_Toc162734619)

[2.4. Ръководство на потребителя 26](#_Toc162734620)

[2.4.1. Крайния потребител 26](#_Toc162734621)

[2.4.2. Потребителя като разработчик 26](#_Toc162734622)

[3. Заключение 26](#_Toc162734623)

[4. Информационни източници 27](#_Toc162734624)

[5. Приложения 28](#_Toc162734625)

# Увод

## Описание

В свят, в който пътуването не е просто лукс, а начин на живот, намирането на идеалното настаняване може да бъде трудна задача. Проектът „BookBnB”, е любителски проект, който има за цел да създаде клонинг на популярната платформа за наемане на жилища Airbnb и да демонстрира използването на различни технологии в уеб разработката. Този проект е не само технологично предизвикателство, но също така предоставя възможност за изследване на различни аспекти на разработката на уеб приложения, включително потребителско изживяване (UX), архитектура на данни и сигурност. В центъра на BookBnB лежи мощен технологичен стек, внимателно подбран спрямо най-новите и иновативни технологии, за да осигури достъп до възможно най-много потребители. Приложението в основата си е базирано на React. Този проект има за цел да предостави уникално изживяване при пътуване за тези, които обичат да изследват естествения свят и да участват в дейности на открито.

## Мотивация

Airbnb революционизира начина, по който хората пътуват и преживяват нови места. Той предлага уникална алтернатива на традиционните хотели, позволявайки на пътуващите да се свързват директно със собствениците на жилища и да резервират престой в целия свят. Като запален пътешественик и почитател на дейностите на открито и приключенията съм използвал Airbnb многократно и винаги съм бил впечатлен от лекотата на използване и разнообразието от опции, които предлага платформата. Но това не означава, че сайта няма своите недостатъци. Въпреки това, като ученик, аз също съм любопитен как работи всичко това от техническа гледна точка. Това ме накара да изследвам възможностите за създаване на клонинг на Airbnb, което би ми позволило да разбера по-добре архитектурата на приложението, да подобря уменията си за разработка на уеб приложения и потенциално да допринеса за развиването му. BookBnB има за цел да реши някои от гореспоменатите проблеми, като позволи на потребителите да търсят и резервират места за настаняване, подходящи за дейности на открито и приключения по целия свят

## Основна цел

Основната цел на този проект е да създаде любителско копие на Airbnb, което позволява на потребителите да разглеждат и резервират места за настаняване по целия свят. Тази версия трябва да включва ключови функции като търсене на място за настаняване, преглед на подробности за имота, резервиране на стая и регистриране на потребител. Което значи, че той също така трябва да поддържа потребителски акаунти, удостоверяване и всички останали функции свързани с един акаунт. Други от основните цели са създаване на ангажиращо потребителско изживяване (UI/UX), което отразява страстта към дейностите на открито и приключенията и осигурява безпроблемно навигиране в платформата. Друга цел на уеб апликацията е да демонстрира използване на много различни и мощни технологии и библиотеки в едно реализирано приложение.

## Очаквания

Очаква се BookBnB да постигне няколко образователни и професионални цели. От една страна, той ще ни помогне да придобием практически опит в разработването на сложни уеб приложения, използвайки съвременни технологии като React, React Hooks, React Router, Firebase и Material UI. Освен това ще ни предоставим възможност да проучим различни аспекти на UX дизайна, архитектурата на данните и сигурността на приложенията. Очакваме също така, че този проект ще ни помогне да демонстрираме нашите умения за разработка на уеб приложения на бъдещи работодатели или клиенти.

## Предизвикателства

Създаването на BookBnB идва с няколко предизвикателства. Едно от най-значимите предизвикателства е проектирането на потребителското изживяване (UI/UX) и интерфейса на приложението, тъй като те трябва да бъдат интуитивни и удобни за потребителя, като същевременно осигуряват функционалности, свързани с дейности на открито и приключения.

Друго предизвикателство е внедряването на сигурна и надеждна система за удостоверяване на потребители. И накрая, интегрирането на различните услуги които използва приложението Firebase. Това може да бъде сравнително сложен процес, изискващ внимателно планиране и тестване, за да се осигури безпроблемна работа.

# Изложение

## Описание на софтуерните технологии

### 2.1.1. Уеб приложение

Уеб приложение, е софтуерно приложение, което е достъпно и управлявано изцяло чрез уеб браузър или по-точно казано през интернет. За разлика от традиционните софтуерни приложения, които изискват инсталиране на устройството на потребителя, уеб приложенията се хостват на отдалечени сървъри и се осъществяват достъп чрез уеб браузър, което ги прави независими от платформата и достъпни от всяко устройство с интернет връзка. Уеб приложенията следват модел на архитектура клиент-сървър, където логиката и данните на приложението се намират на отдалечен сървър (backend), а потребителят взаимодейства с приложението чрез уеб браузър (клиент). Тази архитектура позволява безпроблемна комуникация между клиента и сървъра, като клиентът изпраща заявки до сървъра и получава отговори, съдържащи исканите данни или функционалност. Потребителският интерфейс на уеб приложение се представя на потребителя чрез уеб браузър. Състои се от HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) и JavaScript, които заедно определят структурата, стила и интерактивността на интерфейса на приложението. Съвременните уеб приложения често включват усъвършенствани UI framework-ове и библиотеки като в нашия случай React за създаване на динамични и отзивчиви потребителски интерфейси. Бекендът на уеб приложение се състои от код от страна на сървъра, отговорен за обработката на клиентски заявки, изпълнението на бизнес логиката и взаимодействието с бази данни или външни услуги. Обичайните технологии, използвани в backend разработката, включват езици за програмиране като JavaScript (Node.js) и framework-ове като Express.js, Django, Flask, Rails и Firebase. Бекендът отговаря за задачи като удостоверяване на потребителя, валидиране на данни и операции с бази данни. Уеб приложенията често изискват постоянно съхранение на данни, за да управляват потребителска информация, данни за приложения и други ресурси. Базите данни се използват за ефективно съхраняване и извличане на данни. Популярните избори за бази данни за уеб приложения включват релационни бази данни и бази данни от типа NoSQL като Firebase Firestore. Уеб приложенията разчитат на стандартни комуникационни протоколи като HTTP (Hypertext Transfer Protocol) и HTTPS (HTTP Secure), за да улеснят комуникацията между клиента и сървъра. HTTP се използва за предаване на данни между браузъра и уеб сървъра, докато HTTPS добавя слой на криптиране, за да защити предаването на данни, като гарантира поверителността и целостта на потребителските данни.

Уеб приложенията често трябва да управляват потребителски сесии, състояние на приложението и данни в множество взаимодействия. Техниките за управление на сесии като бисквитки, сесии и токени се използват за поддържане на удостоверяването на потребителя и проследяване на потребителските взаимодействия(Firebase Auth). Освен това модерните уеб приложения могат да използват библиотеки за управление на състоянието от страна на клиента като Redux, MobX или контекстен API в React за управление на сложни състояния на приложения. Веднъж разработени, уеб приложенията трябва да бъдат разположени на уеб сървър, за да станат достъпни през интернет. Внедряването включва конфигуриране на сървърната среда, качване на файлове на приложението и осигуряване на правилна свързаност. Уеб приложенията могат да бъдат хоствани на различни платформи, включително облачни услуги като Firebase Hosting.

### 2.1.2. HTML

HTML (HyperText Markup Language) служи като гръбнакът на световната мрежа, осигурявайки структурата и съдържанието за почти всяка уеб страница, която срещнете онлайн. Това е език за маркиране, който дефинира структурата на уеб документи чрез използване на система от тагове и атрибути за описание на елементите в една страница. В основата си HTML се състои от набор от таг-ове за маркиране, които определят структурата и съдържанието на уеб страница.

Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една еб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Тези тагове са затворени в ъглови скоби (< >) и обикновено идват по двойки - отварящ таг и затварящ таг. Отварящият таг показва началото на елемент, докато затварящият таг обозначава края на този елемент. Някои елементи обаче са самозатварящи се и не изискват затварящ таг. HTML елементите могат да бъдат категоризирани в няколко типа: семантични елементи, елементи на текстово съдържание, мултимедийни елементи, елементи на формуляри и елементи на структурата на документа. HTML документите обикновено са структурирани йерархично, с вложени елементи, образуващи дървовидна структура. Тази йерархична структура е от съществено значение за организиране и стилизиране на съдържанието на уеб страница с помощта на CSS (Cascading Style Sheets) и за добавяне на интерактивност и функционалност с помощта на JavaScript.



**Фигура 1**

### 2.1.3. CSS

CSS или Cascading Style Sheets е основна технология в уеб разработката, която допълва HTML чрез контролиране на представянето и оформлението на уеб документите. Той позволява на разработчиците да стилизират и проектират уеб страници, осигурявайки последователност, естетика и приятно потребителско изживяване на различни устройства и размери на екрана. В основата си CSS се състои от набори от правила, които определят как HTML елементите трябва да се показват в браузъра. Тези набори от правила се състоят от селектори, свойства и стойности. Селекторите са насочени към конкретни HTML елементи, докато свойствата определят визуалните характеристики на тези елементи, като цвят, размер, шрифт, разстояние и позициониране.

Стойностите определят специфичните настройки за всяко свойство. CSS предоставя широк набор от възможности за стилизиране на уеб съдържание, включително селектори, модел на кутия, типография, цветове и фонове, оформление, преходи и анимации и медийни заявки. В обобщение, CSS е критична технология в уеб разработката, която позволява на разработчиците да стилизират и проектират уеб страници, да контролират оформлението и типографията, да прилагат цветове и фонове, да създават адаптивни оформления, да добавят преходи и анимации и да оптимизират потребителското изживяване на различни устройства и размери на екрана. Използвайки силата на CSS, разработчиците могат да създават визуално привлекателни и ангажиращи уеб приложения, които завладяват потребителите и ефективно придават съдържание и идентичност на сайта.

**Фигура 2**



### 2.1.4. JavaScript

JavaScript е многофункционален език за програмиране, който обикновено се използва за създаване на интерактивно и динамично уеб съдържание. Като основен компонент на уеб разработката, JavaScript позволява на разработчиците да добавят функционалност, да манипулират данни и да реагират на потребителските взаимодействия в уеб приложенията. JavaScript се използва предимно за скриптове от страна на клиента, което означава, че работи в уеб браузъра на потребителя, а не на отдалечен сървър. Това позволява на JavaScript да взаимодейства с Document Object Model (DOM), който представлява структурата на HTML и XML документи. Чрез достъп до и модифициране на DOM, JavaScript може динамично да актуализира съдържанието и поведението на уеб страниците в отговор на потребителски действия, като кликвания, събития за въвеждане и събития за зареждане на страници.

Ключовите характеристики и възможности на JavaScript включват променливи и типове данни, оператори и изрази, оператори на потока на управление, функции, обекти и обектно-ориентирано програмиране (ООП), обработка на събития, асинхронно програмиране, обработка на грешки, модули и модуларизация и API на браузъра.

A yellow and black logo

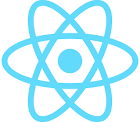
Description automatically generated В обобщение, JavaScript е мощен език за програмиране, който позволява на разработчиците да създават интерактивни и динамични уеб приложения чрез манипулиране на DOM, обработка на потребителски взаимодействия, извършване на асинхронни задачи и достъп до API на браузъра. Със своята гъвкавост и широк набор от възможности, JavaScript играе централна роля в модерното уеб развитие, като дава възможност на разработчиците да изграждат ангажиращи и отзивчиви уеб изживявания.

**Фигура 3**

### 2.1.5. React

React е JavaScript библиотека за изграждане на потребителски интерфейси, разработена и поддържана от Facebook. Тя се използва широко в уеб разработката за създаване на интерактивни и динамични приложения и потребителски интерфейси за уеб приложения, както в този проект. React използва архитектура, базирана на компоненти, позволявайки на разработчиците да създават многократно използваеми UI компоненти, които капсулират логиката и представянето на различни части на приложението. Приложенията на React са изградени с помощта на компоненти, които са многократно използвани и независими градивни елементи на потребителския интерфейс. Компонентите могат да бъдат прости, представляващи един UI елемент, или сложни, капсулиращи множество UI елементи и функционалност. Компонентите могат да бъдат композирани заедно, за да се създадат по-големи, по-сложни потребителски интерфейси. Компонентите на React могат да имат състояние, което представлява данни, които могат да се променят с течение на времето. Състоянието се управлява вътрешно в рамките на компонент и може да се променя с помощта на hook-а useState.

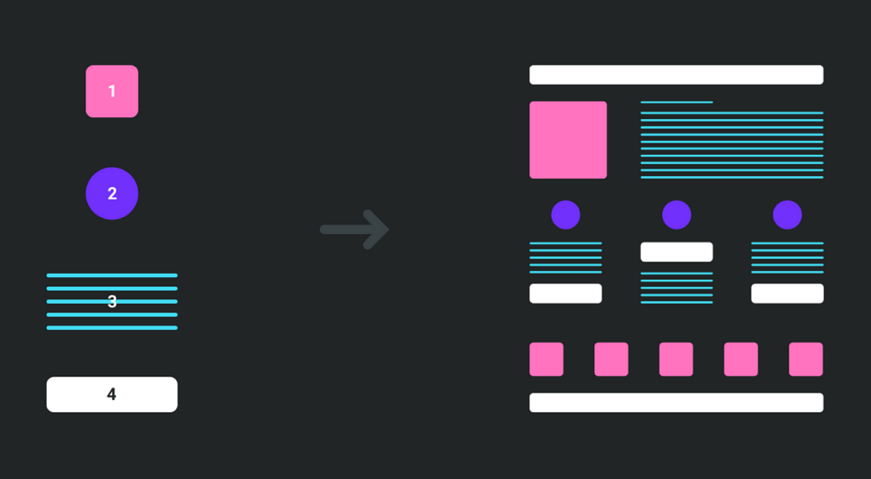
Props (съкратено от свойства) се използват за предаване на данни от родителски към дъщерни компоненти, което позволява комуникация между компонентите. Компонентите на React имат свойства за жизнения цикъл, които позволяват на разработчиците да се включат в различни етапи от жизнения цикъл на компонента, като инициализация, монтиране, актуализиране и демонтиране. Методите на жизнения цикъл предоставят възможности за изпълнение на задачи като извличане на данни, настройка на събития и почистване на ресурси.

React несъзнателно насърчава разработката, базирана на компоненти, позволявайки на разработчиците да създават модулни и повторно използвани UI компоненти. Чрез композиране на компоненти заедно, разработчиците могат да изградят сложни потребителски интерфейси, като същевременно поддържат разделяне на проблемите и организация на кода. В обобщение, React е мощна и ефективна библиотека за изграждане на потребителски интерфейси в уеб приложения. Със своята базирана на компоненти архитектура и голям набор от функции, React предоставя на разработчиците инструментите, от които се нуждаят, за да създадат интерактивни, ефективни и поддържаеми потребителски интерфейси.

**Фигура 4**

### 2.1.6. React Components

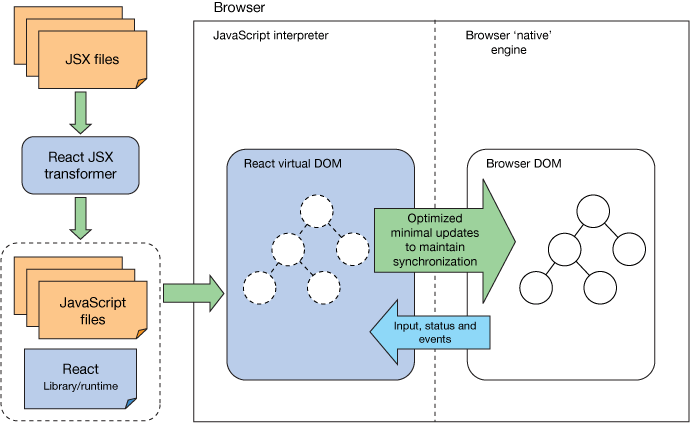
Компонентите на React са градивните елементи на React приложение, капсулирайки както потребителския интерфейс, така и функционалността на различни части на приложението. Те позволяват на разработчиците да създават многократно използвани и модулни части от потребителски интерфейс, които могат да бъдат съставени заедно, за да изградят сложни потребителски интерфейси. Компонентът на React може да се разглежда като JavaScript функция или клас, който връща част от JSX (JavaScript XML), който описва как компонентът трябва да се визуализира в браузъра. Компонентите могат да представляват всичко - от прости елементи като бутони или полета за въвеждане до по-сложни елементи на потребителския интерфейс като навигационни ленти, формуляри или цели секции от уеб страница. Има два основни типа компоненти в React: функционални компоненти и компоненти на класа. Функционалните компоненти са функции на JavaScript, които приемат подпори (съкратено от свойства) като вход и връщат JSX, за да опишат потребителския интерфейс. Те обикновено се използват за компоненти без състояние или презентационни компоненти, където поведението на компонента се определя единствено от неговите подпори. Компонентите на класа са JavaScript класове, които разширяват класа React.Component. Те имат метод render(), който връща JSX, описвайки как компонентът трябва да се визуализира. Компонентите на класа могат също да имат състояние, методи на жизнения цикъл и други функции. В допълнение към тези основни типове, React поддържа и други типове компоненти, като Pure Components, Higher-Order Components (HOC) и Render Props.



**Фигура 5**

### 2.1.7. JSX (JavaScript XML)

JSX или JavaScript XML е разширение на синтаксиса на JavaScript, което позволява писането на HTML-подобен код в JavaScript. С JSX разработчиците могат да пишат HTML-подобен код директно в своите JavaScript файлове, което улеснява визуализирането и разбирането на структурата на потребителския интерфейс. JSX кодът наподобява HTML, но може да включва и JavaScript изрази и логика, което позволява динамични и интерактивни потребителски интерфейси. JSX елементите представляват градивните елементи на потребителския интерфейс на приложението React. Тези елементи могат да бъдат прости, представляващи основни HTML елементи като div, spans или параграфи, или могат да бъдат по-сложни, представляващи потребителски React компоненти. JSX също поддържа вграждане на JavaScript изрази във фигурни скоби {}, което позволява на разработчиците динамично да генерират съдържание, да изчисляват стойности или да изпълняват логика в JSX елементи. Елементите на JSX могат също да съдържат атрибути, подобни на HTML елементите, посочени с помощта на синтаксис, подобен на HTML, и включително манипулатори на събития, вградени стилове и други подпори, предадени на компонентите на React.



**Фигура 6**

### 2.1.8. Virtual DOM

Виртуалният DOM (Document Object Model) е ключова концепция в React, която играе решаваща роля в оптимизирането на производителността на React приложенията.

При традиционната уеб разработка, когато се правят промени в потребителския интерфейс, DOM на браузъра се актуализира директно, карайки браузъра да изобрази отново засегнатите елементи. Този процес обаче може да бъде неефективен, особено когато се работи със сложни потребителски интерфейси или чести актуализации, тъй като изисква браузърът да изобрази отново цялото DOM дърво, дори ако само малка част от него е променена.

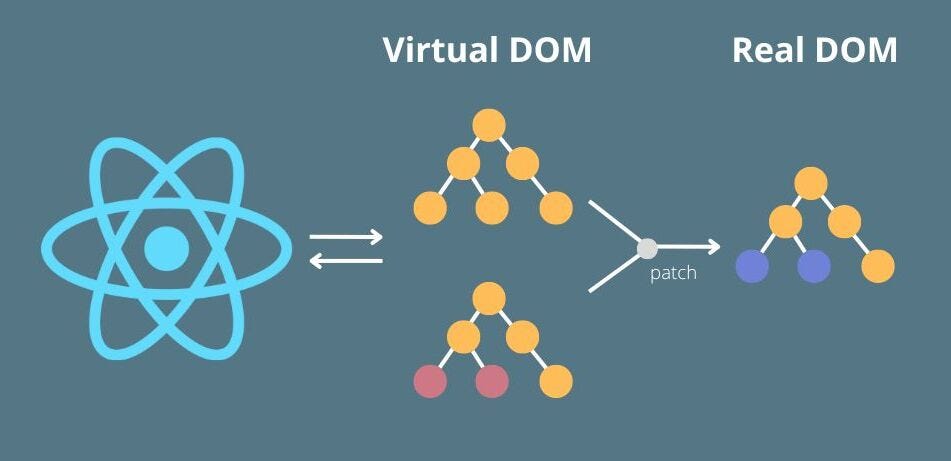
За да се справи с този проблем, React въвежда концепцията за виртуалния DOM. Виртуалният DOM е леко представяне в паметта на действителния DOM. Когато се направят промени в потребителския интерфейс в приложение на React, вместо да актуализира незабавно действителния DOM, React актуализира първо виртуалния DOM.

След това React сравнява актуализирания виртуален DOM с предишната версия, за да определи минималния набор от промени, необходими за синхронизиране на действителния DOM с виртуалния DOM. Този процес е известен като помирение.

След като React определи необходимите промени, той актуализира действителния DOM ефективно, като модифицира само специфичните елементи, които са се променили. Този подход значително подобрява производителността и намалява количеството работа, изисквано от браузъра, което води до по-бързо изобразяване и по-плавно потребителско изживяване.

Виртуалният DOM в React служи като абстракционен слой, който позволява на разработчиците да работят с опростено представяне на DOM, докато React се справя с оптимизирането на ефективното актуализиране на действителния DOM.

**Фигура 7**



### 2.1.9. React Hooks

React Hooks са функция, въведена в React 16.8, която позволява на функционалните компоненти да имат достъп до състояние и други функции на React, без да е необходимо да се напише клас. Те предоставят по-кратък и изразителен начин за управление на логиката със състояние във функционални компоненти, което улеснява споделянето на логика между компонентите и намалява сложността на базираните на клас компоненти.

Hook-овете са функции, които позволяват на разработчиците да се „закачат“ за функции на React като състояние, методи на жизнения цикъл, контекст и др. Някои често използвани кукички включват useState, useEffect, useContext и useCallback.

Hook-а позволява на функционалните компоненти да управляват състоянието чрез предоставяне на променлива на състоянието и функция за актуализиране на тази променлива на състоянието. Това елиминира нуждата от компоненти на клас и this.setState().

Hook-а useEffect позволява на функционалните компоненти да изпълняват странични ефекти, като извличане на данни, абонаменти или DOM манипулации. Той замества методи на жизнения цикъл като componentDidMount, componentDidUpdate и componentWillUnmount.

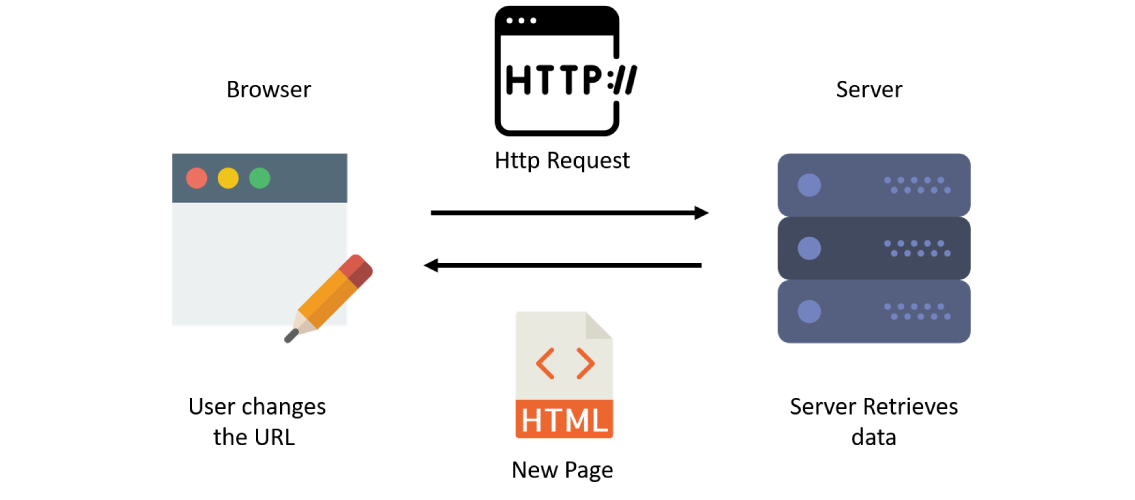
Hook-а useContext осигурява начин за използване на контекст във функционални компоненти, позволявайки на компонентите да имат достъп до споделени данни, без да използват пробиване на подпори.

Като цяло React Hooks предлагат по-модерен и гъвкав подход за изграждане на React компоненти.

### 2.1.10. React Router

React Router е популярна библиотека, използвана за обработка на маршрутизиране в React приложения. Той позволява на разработчиците да създават приложения с една страница с множество изгледи или страници, което позволява на потребителите да навигират между различни части на приложението, без да причиняват пълно презареждане на страницата. React Router предоставя декларативен начин за дефиниране на маршрути и компоненти за свързване, което улеснява създаването на навигационна структура за приложението. Той използва подход, базиран на компоненти, където всеки маршрут е представен от React компонент. Основните компоненти, предоставени от React Router, включват BrowserRouter, Route, Switch, Link и Redirect. BrowserRouter използва API за история на HTML5, за да поддържа потребителския интерфейс в синхрон с URL адреса и осигурява основата за маршрутизиране. Маршрутът изобразява потребителския интерфейс въз основа на текущия URL адрес, което позволява на разработчиците да определят кой компонент да се изобразява за даден URL път. Превключвателят изобразява първия дъщерен маршрут или пренасочване, който съответства на текущия URL адрес, полезен за изобразяване само на единичен маршрут. Link предоставя начин за навигация между маршрути в приложението. Пренасочването изобразява ново местоположение на маршрута, когато пътят съвпада. React Router също поддържа разширени функции като вложени маршрути, параметри на маршрута, защита на маршрута и други, което го прави подходящ за изграждане на сложни навигационни структури и потребителски интерфейси.

**Фигура 8**



### 2.1.11. Material UI

Material-UI е популярен React framework, която предоставя набор от адаптивни и отзивчиви компоненти, следвайки указанията на Google за материален дизайн. Той опростява процеса на изграждане на потребителски интерфейси в React приложения, като предлага широка гама от предварително проектирани компоненти, стилове и теми.

Material-UI компонентите са проектирани да бъдат силно модулни и многократно използвани, което улеснява разработчиците да създават последователни и визуално привлекателни потребителски интерфейси. Тези компоненти включват бутони, формуляри, навигационни ленти, карти, модали, подсказки и други, покриващи широк набор от често срещани модели на потребителския интерфейс.

Една от ключовите характеристики на Material-UI е неговата тематична система, която позволява на разработчиците да персонализират външния вид на компонентите, за да отговарят на изискванията за брандиране и дизайн на техните приложения. Material-UI предоставя тема по подразбиране с конфигурируеми опции за цветове, типография, интервали и други дизайнерски токени. Разработчиците могат също така да създават персонализирани теми или да заменят конкретни стилове, използвайки CSS-in-JS техники.

Material-UI следва съвременните най-добри практики за уеб разработка, като достъпност и адаптивен дизайн, като гарантира, че приложенията, създадени с Material-UI, са достъпни за всички потребители и работят добре на различни устройства и размери на екрана.

В допълнение към своите основни компоненти и тематична система, Material-UI предлага допълнителни помощни програми и инструменти за подобряване на изживяването при разработката, включително икони, варианти на типография, преходи и компоненти за оформление.

**Фигура 9**



### 2.1.12. Framer Motion

Framer Motion е популярна библиотека за анимации за приложения на React, която позволява на разработчиците да създават зашеметяващи и плавни анимации с лекота. Той предоставя прост и интуитивен API за добавяне на анимации към компонентите на React, включително преходи, жестове и сложни анимации.

Framer Motion предлага няколко ключови функции, които го правят мощен инструмент за създаване на динамични и интерактивни потребителски интерфейси.

Първо, той използва декларативен API, позволяващ на разработчиците да дефинират анимации, използвайки прост синтаксис и интуитивни подпори, което улеснява разбирането и поддържането на сложни анимации в компонентите на React.

Второ, Framer Motion въвежда концепцията за варианти, която позволява на разработчиците да дефинират различни състояния на анимация и плавен преход между тях. Вариантите могат да анимират свойства като позиция, мащаб, ротация, непрозрачност и други.

Трето, Framer Motion осигурява вградена поддръжка за анимации на оформлението, позволявайки по-плавни преходи между различни състояния на UI компонент чрез анимиране на промени в свойствата на оформлението като ширина, височина и позиция. Четвърто, включва поддръжка за жестове, което позволява на разработчиците да създават интерактивни анимации, задействани от потребителско въвеждане, като жестове за плъзгане, плъзгане, щипване и завъртане. Това позволява на разработчиците да създават ангажиращи и интуитивни потребителски интерфейси, които реагират на взаимодействие с докосване и мишка.

И накрая, Framer Motion е съвместим с изобразяване от страна на сървъра (SSR) и отложено зареждане, което го прави подходящ за изграждане на производителни и достъпни уеб приложения, които работят безпроблемно на различни платформи и устройства. Като цяло, Framer Motion е многофункционална и мощна анимационна библиотека за React приложения, предоставяща на разработчиците необходимите инструменти за създаване на плавни, интерактивни и визуално зашеметяващи потребителски интерфейси.

**Фигура 10**



### 2.1.13. Мениджър на пакетите

Мениджърът на пакети е софтуерен инструмент, който помага при управлението, инсталирането и премахването на софтуерни пакети или библиотеки в рамките на компютърна система или специфична среда за програмиране. Той опростява процеса на обработка на зависимости, като гарантира, че всички необходими компоненти или модули са налични и правилно интегрирани за гладкото функциониране на приложенията или проектите.

Мениджърите на пакети са особено важни при програмирането и разработката на софтуер, където проектите често разчитат на различни външни библиотеки или модули. Те рационализират процеса на придобиване и управление на тези зависимости, като улесняват разработчиците да се съсредоточат върху основните си задачи, вместо да боравят ръчно с отделни пакети. BookBnB използва NPM като мениджър на пакетите.

NPM (Node Package Manager) е мениджър на пакети за JavaScript и мениджър на пакети по подразбиране за Node.js, среда за изпълнение, която позволява изпълнение на JavaScript код извън уеб браузър. Той служи като централно хранилище за повторно използваем JavaScript код, предоставяйки лесен начин за разработчиците да споделят, откриват и управляват зависимости в своите проекти.

С NPM разработчиците могат лесно да инсталират, актуализират и премахват пакети (известни също като модули или библиотеки), необходими за техните приложения. Тези пакети могат да варират от малки помощни функции до цялостни рамки, предоставящи решения за различни аспекти на разработката на JavaScript.

**Фигура 11**



### 2.1.14. Firebase

Основата на backend-а(автентикация, запазване на снимки, и др.) на проекта е базирана на Firebase. Firebase е цялостна платформа, предоставена от Google, която предлага разнообразие от услуги за бързо и ефективно изграждане на уеб и мобилни приложения. Той предоставя широк набор от инструменти и функции, включително услуги за база данни, удостоверяване, облачно съхранение, хостинг и други, всички интегрирани в една платформа.

Firebase Cloud Firestore е гъвкава и мащабируема NoSQL услуга за база данни, предоставена от Firebase. Той позволява на разработчиците да съхраняват и синхронизират данни в реално време между клиенти и сървъри, което го прави идеален за изграждане на приложения в реално време като приложения за чат, инструменти за сътрудничество и табла за управление на данни на живо. Firestore поддържа мощни възможности за заявки и индексиране, което позволява на разработчиците да извличат и манипулират данни ефективно.

Firebase Cloud Storage е базирана на облак услуга за съхранение, предоставена от Firebase, която позволява на разработчиците да съхраняват и обслужват генерирано от потребителите съдържание, като изображения, видеоклипове и документи, сигурно и надеждно. Той предоставя мащабируеми решения за съхранение с вградени функции за сигурност, включително контрол на достъпа и криптиране, което го прави подходящ за съхранение и обслужване на мултимедийно съдържание в уеб и мобилни приложения.

Firebase Authentication е услуга, предоставена от Firebase, която позволява на разработчиците да удостоверяват потребители и да управляват самоличността на потребителите в техните приложения. Той поддържа различни методи за удостоверяване, включително имейл/парола, социално влизане (напр. Google, Facebook, Twitter), удостоверяване по телефона и персонализирано удостоверяване, което позволява на разработчиците да изберат метода за удостоверяване, който най-добре отговаря на изискванията на тяхното приложение. Firebase Authentication също така предоставя функции като управление на потребители, потребителски роли и многофакторно удостоверяване (MFA) за подобрена сигурност

A screenshot of a computer

Description automatically generated

**Фигура 12**

### 2.1.15. Използвани пакети

Проекта включва сбор от няколко пакета с помощта на които имаме готова функционалност за библиотеките.

**Фигура 13**

A screen shot of a computer program

Description automatically generated

По важните от тях са пакетите:

* Firebase - NPM пакетът Firebase е официалната клиентска библиотека на JavaScript за Firebase, която е цялостна платформа за разработка, предоставена от Google. Той позволява на разработчиците лесно да интегрират услугите на Firebase в своите JavaScript проекти, включително уеб и Node.js приложения. Пакетът предоставя унифициран API и набор от функции, които позволяват на разработчиците да взаимодействат с различни услуги на Firebase
* React – NPM пакетът react е официалният пакет, който предоставя основната функционалност на библиотеката React за изграждане на потребителски интерфейси в приложения на JavaScript. React е популярна JavaScript библиотека, разработена от Facebook и широко използвана за създаване на динамични и интерактивни UI компоненти.
* React-dom - React DOM NPM Пакетът е официалният пакет, който осигурява интеграцията между React и Document Object Model (DOM) на браузъра.
* React Router Dom - Пакетът е JavaScript библиотека, която предоставя възможности за маршрутизиране за React приложения. Това е популярно разширение на библиотеката на React Router, специално проектирано за уеб приложения, които използват React за изграждане на потребителски интерфейси.
* @mui/material - пакетът е част от библиотеката Material-UI, който е популярен React UI framework, който прилага насоките на Google за материален дизайн. Material-UI предоставя набор от адаптивни и отзивчиви компоненти за изграждане на модерни уеб приложения с последователен и визуално привлекателен дизайн.

## Реализация на приложението

Приложението и неговите функционалности макар и семпли дават много добра идея за силата на уеб технологиите и предоставените от тях възможности. В едно такова приложение с отворен код се дават нагледни примери за front-end технологии и разработка. Работа с backend, сесии, бази данни и автентикация на потребители. Проектът служи за една добра основа за разработчиците в уеб среда.

## Поставени цели за проекта

* Намиране на актуална тема и нуждата от подобен софтуер
* Избор на рентабилни и лесни за поддръжка технологии
* Избор на back end за приложението (лесен начин за съхраняване на данни като чатове снимки и контакти)
* Избор на сигурни и защитени начини за регистриране на потребителите
* Запознаване с динамичен уеб дизайн
* Проектиране на динамичен уеб дизайн
* Публикуване на приложението

## Ръководство на потребителя

### 2.4.1. Крайния потребител

- Посетете сайта на BookBnB

- Изберете една от трите опции : Регистриране, влизане в профил или продължете като гост

- Попълнете поисканата от вас информация и продължете към началната страница

- От там може да потърсите различни места за резервиране, да направите резервация и да редактирате профила си (Погледнете приложените снимки по долу)

### 2.4.2. Потребителя като разработчик

Понеже BookBnB е платформа с отворен код, всеки един потребител който иска да се запознае с кода може да го инсталира и да го настрои за неговите нужди.

* След инсталиране на проекта от Гитхъб репозиторията и отварянето и в подходяща среда за разработка трябва да се инсталират нужните npm пакети от мениджъра на пакети с командата „npm install“. Командата трябва да бъде извикана от терминала в директорията в която се намира package-lock.json и package.json файловете, за да може мениджърът да намери нужните пакети и кои версии на тези пакети трябва да бъдат инсталирани.
* След всичко това за най-лесно пускане на проекта може да се използва “npm start” командата за пускане на приложението на локален сървър на лаптопа(в тестова версия)

# Заключение

В заключение, проектът BookBnB представлява цялостно и удобно решение за пътници, търсещи настаняване по целия свят. Като използва силата на React, Firebase, Material-UI и Framer Motion, BookBnB предлага интуитивно изживяване за откриване, резервиране и управление на места за настаняване.

Със своята основна гама от функции, включително синхронизиране на данни в реално време с Firebase Cloud Firestore, сигурно удостоверяване с Firebase Auth и плавни анимации с Framer Motion, BookBnB поставя нов стандарт за модерни уеб приложения в хотелиерската индустрия.

Освен това предоставеното ръководство за потребителя гарантира, че потребителите могат да навигират в платформата без усилие, което прави изживяването им при резервация приятно и без стрес. Като дава приоритет на потребителското изживяване, достъпността и функционалността, BookBnB има за цел да революционизира начина, по който пътниците намират и резервират места за настаняване, като в крайна сметка подобрява изживяването им при пътуване и насърчава незабравими пътувания.

В допълнение, BookBnB е с отворен код, което позволява на други разработчици да използват неговата кодова база като основа за своите проекти. Това демонстрира ангажимента на проекта към сътрудничество, прозрачност и развитие, ръководено от общността. Като споделя своя изходен код и демонстрира използването на съвременни технологии, BookBnB вдъхновява иновациите и предоставя ценна информация за най-добрите практики за изграждане на авангардни уеб приложения.

В обобщение, проектът BookBnB е пример за иновация, ефективност и ориентиран към потребителя дизайн, като се позиционира като водеща платформа в конкурентния пейзаж на онлайн резервациите за настаняване. С непрекъснати актуализации, подобрения и обратна връзка от потребителите, BookBnB е готов да предефинира бъдещето на резервирането на пътувания и да вдъхнови проучването по целия свят.

# Информационни източници

[**https://en.wikipedia.org/wiki/React\_(software)**](https://en.wikipedia.org/wiki/React_(software))

[**React.js Wiki - Preguntas típicas con respuesta y soluciónreactjs.wikihttps://**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://www.reactjs.wiki/&ved=2ahUKEwjBkvjThZ2FAxWCSvEDHRRRDjIQFnoECBMQAQ&usg=AOvVaw20911fDlIFUgEymetvK561)[**www.reactjs.wiki**](http://www.reactjs.wiki)

[**npm Docsnpm**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://docs.npmjs.com/&ved=2ahUKEwj1xpT5hZ2FAxU4A9sEHcL-CjwQFnoECBAQAQ&usg=AOvVaw1vq7wii2AFCthGAtuQVBfs) [**Docshttps://docs.npmjs.com**](Docshttps://docs.npmjs.com)

[**HTML: HyperText Markup Language - MDN Web DocsMDN Web Docshttps://developer.mozilla.org › en-US › Web › HTML**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://developer.mozilla.org/en-US/docs/Web/HTML&ved=2ahUKEwjJ85L-hZ2FAxWTBdsEHYhqBcwQFnoECBUQAQ&usg=AOvVaw23zU8Os1zIEwrE-SW_k5x4)

[**CSS: Cascading Style Sheets - MDN Web DocsMDN Web Docshttps://developer.mozilla.org › en-US › Web › CSS**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://developer.mozilla.org/en-US/docs/Web/CSS&ved=2ahUKEwjP3ZSEhp2FAxUjX_EDHZHaB74QFnoECBUQAQ&usg=AOvVaw2uGFFuZmjPce8PuYb0p8Tp)

[**CSS: Cascading Style Sheets - MDN Web DocsMDN Web Docshttps://developer.mozilla.org › en-US › Web › CSS**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://developer.mozilla.org/en-US/docs/Web/CSS&ved=2ahUKEwjP3ZSEhp2FAxUjX_EDHZHaB74QFnoECBUQAQ&usg=AOvVaw2uGFFuZmjPce8PuYb0p8Tp)

[**JavaScript - MDN Web Docs - MozillaMozillahttps://developer.mozilla.org › en-US › Web › JavaScript**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://developer.mozilla.org/en-US/docs/Web/JavaScript&ved=2ahUKEwi_4N-Zhp2FAxVqRfEDHevWB-MQFnoECAYQAQ&usg=AOvVaw1Il_CfTbNi4CXc-0nBN5rP)

[**Firebase DocumentationGooglehttps://firebase.google.com › docs**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://firebase.google.com/docs&ved=2ahUKEwjQmvOghp2FAxXHYPEDHW9vAbQQFnoECAcQAQ&usg=AOvVaw0PvZiBaWaogN0S0Q7uY2lT)

[**Documentation | Framer for DevelopersFramerhttps://www.framer.com › motion**](https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://www.framer.com/motion/&ved=2ahUKEwj9vLuwhp2FAxVsRPEDHYgaBysQFnoECA4QAQ&usg=AOvVaw2RIk4bM_mCKCOVNSf5cZ0B)

# A screenshot of a login form Description automatically generatedA screenshot of a login form Description automatically generated Приложения

Фигура 15

Фигура 14

Фигура 16

A screenshot of a computer screen

Description automatically generated

Фигура 17

A screenshot of a hotel

Description automatically generatedA screenshot of a computer screen

Description automatically generated

Фигура 18

A screenshot of a computer

Description automatically generated

Фигура 19