



الجامعة الافتراضية السورية  
SYRIAN VIRTUAL UNIVERSITY

الجامعة الافتراضية السورية  
برنامج الهندسة المعلوماتية ITE

ITE\_BPR601\_C33\_S23

# تصميم موقع للحجز في رحلة

TravelAid ✈

إعداد الطلاب:

ياسر الحمد yaser\_122441

عبد الرحمن الأوتاني abd\_alrhman\_108964

آلاء الحججي Alaa\_128963

## مقدمة

يهدف هذا المشروع الى تأمين واجهة الكترونية تسهل على المستخدم عمليات الاستعراض والبحث والاختيار والحجز، من بين مجموعة من الرحلات الى مختلف الوجهات السياحية في جميع أنحاء العالم. عن طريق بنية برمجية تم بناؤها بواسطة أحدث التقنيات البرمجية في سنة 2024. لتقديم بيئة برمجية متكاملة سواءً من جهة واجهة بسيطة سهلة الاستخدام ، أو من جهة السرعة البالغة في الاداء تناسباً مع أوسع مجال من التنوع في قدرات جهاز المستخدم النهائي، أو من جهة بنية تحتية سليمة و شاملة قادرة على استيعاب متطلبات النظام بشكل كامل.

سنستعرض في هذا التقرير مراحل بناء المشروع، والتقنيات التي تم بناؤه بواسطتها، وسبب اختيار كل منها، وطريقة تشغيل المشروع.

## 1- الميزات الرئيسية لموقع TravelAid:

### 1. تسجيل الدخول وإنشاء الحسابات:

يتيح الموقع للمستخدمين تسجيل الدخول إلى حساباتهم الموجودة أو إنشاء حسابات جديدة للوصول إلى الميزات الكاملة للموقع. حيث يمكن لمن يزور الموقع لأول مرة تصفح الرحلات والدخول إلى تفاصيل الرحلات ولكن لن يتمكن من حجز الرحلة التي يرغب بها إلا إذا قام بتسجيل الدخول. **فعندما يضغط على زر:**

Reserve

ستظهر له **النافذة** التالية لتوجهه إلى عملية تسجيل الدخول:

×

Register

**Welcome to TravelAid**



Create an account!

Continue

[Already have an account? Log in](#)

## 2. استعراض وحجز الرحلات:

يمكن للمستخدمين بعد ان أتموا عملية تسجيل الدخول استعراض الرحلات المتاحة بسهولة والتفاعل معها، ومن ثم حجز الرحلات التي يفضلونها بنقرة واحدة.



Beach

Modern

Countryside

Pools

Lake

Skating


Castles

Caves


Camping

Arctic


Desert




Asia, United Arab Emirates  
Desert  
\$ 235 night




Africa, Morocco  
Desert  
\$ 139 night




Antarctic, French Southern and Antarctic Lands  
Arctic  
\$ 655 night




Americas, Canada  
Camping  
\$ 112 night




Americas, United States  
Camping  
\$ 230 night




Asia, Jordan  
Caves  
\$ 108 night




Asia, Turkey




Europe, Ireland




Europe, United Kingdom



Americas, United States



Europe, Switzerland



Europe, Belarus

عند الضغط على اي رحلة سينتقل المستخدم الى صفحة تفاصيل الرحلة، حيث يظهر فيها تفاصيل مكان الإقامة من عدد الاشخاص وعدد الغرف وعدد الحمامات بالإضافة الى سعر الليلة الواحدة ضمن مكان الإقامة هذا.

### Desert Mirage Retreat

Asia, United Arab Emirates



3 guests 2 rooms 2 bathrooms

\$ 235 night

بالإضافة الى شرح يصف هذه الرحلة مع وجود عنصر اختيار مدة الرحلة وزر الحجز و عنصر يحدد بدقة مكان هذه الرحلة على الخريطة.



#### Desert

This property is in the desert!

Discover the allure of the Arabian Desert at Desert Mirage Retreat, nestled amidst the mystical dunes of Dubai, United Arab Emirates. Immerse yourself in luxury as you unwind by our infinity pool, gazing out upon the undulating sands and shimmering skyline. Experience the thrill of desert adventures with dune bashing, sandboarding, and camel riding excursions, or simply savor the tranquility of the desert sunset from the comfort of our exclusive desert camp. Indulge in gourmet dining experiences showcasing the finest Middle Eastern cuisine, complemented by the warm hospitality of our desert guides. With lavish accommodations, world-class amenities, and the captivating beauty of the Arabian Desert, Desert Mirage Retreat offers an unforgettable escape into the opulent realm of desert luxury.

April

2024

Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4

Reserve

Total

\$ 235



### 3. إدارة الحجوزات الشخصية:

حيث يمكن للمستخدم بعد ان اتم عملية حجز الرحلة الانتقال الى صفحة الحجوزات حيث يستطيع ان يستعرض كافة الرحلات التي قام بحجزها ويستطيع ان يلغيها بضغطة زر واحدة:

#### Trips

Your Reservations



Europe, Belarus

Apr 17, 2024 - Apr 17, 2024

\$ 163

Cancel reservation



Africa, Morocco

Apr 17, 2024 - Apr 17, 2024

\$ 139

Cancel reservation



Asia, United Arab Emirates

Apr 15, 2024 - Apr 15, 2024

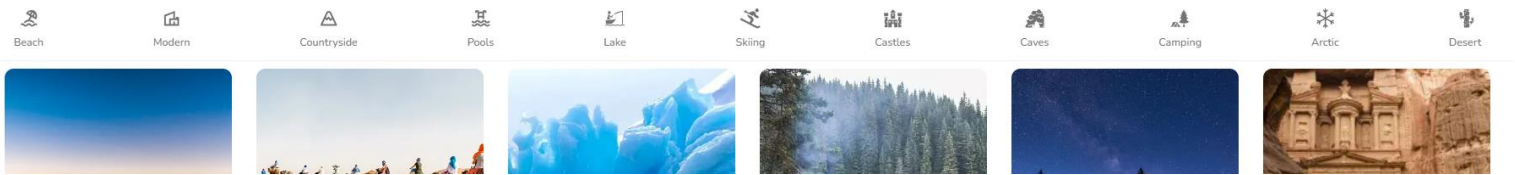
\$ 235

Cancel reservation

#### 4. تجربة مستخدم متميزة:

يهدف المشروع إلى توفير تجربة مستخدم سلسة ومريحة، من خلال استخدام تقنيات حديثة ومكتبات متقدمة وتصميم مرن وجذاب.

تجربة المستخدم المميزة في TravelAid تتميز بعدة عوامل تسهم في جعل تفاعل المستخدمين مع الموقع تجربة فريدة وممتعة. بدءًا من تنوع الوجهات السياحية المقدمة وصولاً إلى واجهة المستخدم السلسة وسهولة استعراض وتصفية الرحلات حسب نوع الرحلة سواء كانت على البحر أو في الريف أو كانت رحلة للتزلج على الثلج. وكل ذلك يتم عبر عنصر واحد فقط:



عندما يتفاعل المستخدمون مع الموقع، يتم استقبالهم بواجهة مستخدم بسيطة وجذابة، تعرض بشكل واضح الرحلات المتاحة والوجهات السياحية المثيرة للاهتمام. باستخدام تقنيات مثل ReactJS و Next.js، يتم تحقيق هذه الواجهة بسرعة وكفاءة، مما يسمح بتحميل سريع للصفحات وتجربة تفاعلية سلسة للمستخدمين. دور Next.js في تجربة المستخدم يظهر بوضوح في تحسين أداء الموقع وسلاسة التنقل بين الصفحات. بفضل تقنيته في التحميل التلقائي والمسبق للصفحات، يمكن للمستخدمين التنقل بين الصفحات بسرعة وبدون تأخير، مما يعزز تجربة التصفح.

بالإضافة إلى ذلك، يساهم استخدام Typescript في زيادة موثوقية التطبيق وسهولة صيانه.

تتيح لغة Typescript للمطورين كتابة كود أكثر دقة ووضوح حيث اضافت هذه اللغة مفهوم الأنماط على لغة JavaScript، مما يساعد في تحديد الأخطاء المحتملة قبل وقوعها، مما يؤدي إلى تقليل أخطاء التطبيق وزيادة كفاءة التطوير.

بالنهاية، يجمع TravelAid بين تصميم متقن وتقنيات متطورة لتقديم تجربة مستخدم استثنائية. من خلال الجمع بين الوظائف المميزة والأداء العالي والتصميم الجذاب،



### 1. ReactJS:

- تم اختيار ReactJS لعدة أسباب تجعله الخيار المثالي لبناء واجهة المستخدم الديناميكية والتفاعلية:
1. **تجربة المطور الرائعة:** يوفر ReactJS تجربة تطوير ممتعة وفعالة للمطورين. بفضل بنيته الواضحة والمنظمة، يمكن للمطورين بسهولة فهم كيفية بناء وتنظيم تطبيقاتهم مع ReactJS.
  2. **إعادة الاستخدام السهلة Reusability:** يسمح نهج ReactJS لإعادة استخدام المكونات بسهولة، مما يوفر الكفاءة في عملية التطوير ويقلل من حجم الشفرة المكررة.
  3. **الأداء العالي:** يُعتبر ReactJS من بين أفضل مكتبات JavaScript لأداء واجهة المستخدم UI ، حيث يتميز بالسرعة والكفاءة في تحديث العناصر وإعادة رسمها، مما يؤدي إلى تجربة مستخدم سلسة ومريحة.
  4. **تكاملاً ممتازاً مع مكتبات أخرى:** يتيح ReactJS تكاملاً ممتازاً مع مجموعة واسعة من المكتبات والأدوات الخارجية، مما يتيح للمطورين استخدام المكتبات المفضلة لديهم لتعزيز وظائف التطبيق.
  5. **دعم للتطبيقات الكبيرة والمعقدة:** يمكن ل ReactJS التعامل مع تطبيقات الويب الكبيرة والمعقدة بكفاءة، مما يجعله الخيار المثالي لبناء TravelAid ، وهو تطبيق يتضمن العديد من الميزات والوظائف المتطلبة.

### 2. Typescript + Next.js 14:

تم اختيار Next.js لعدة أسباب مهمة منها:

- a. **تحسين أداء التطبيق:** يعتبر Next.js معروفاً بأدائه العالي، حيث يوفر تجربة تصفح سلسة وسريعة للمستخدمين. يعمل Next.js على توليد صفحات الويب ديناميكياً مع تحميل سريع، مما يقلل من زمن التحميل ويعزز سرعة التفاعل.
- b. **SEO تحسين محركات البحث:** تقدم Next.js تجربة ممتازة فيما يتعلق بتحسين محركات البحث، حيث تدعم توليد الصفحات الثابتة مسبقاً (SSG) وتوليد الصفحات ديناميكياً (SSR) ، مما يساعد في تحسين ترتيب الموقع في نتائج البحث.
- c. **دعم للتطبيقات الديناميكية:** تتيح Next.js للمطورين بناء تطبيقات ويب ديناميكية مع إمكانية توليد الصفحات على الخادم (SSR) أو جانب العميل (CSR) أو التوليد الساكن (SSG) ، مما يتيح لهم اختيار الأسلوب الأمثل وفقاً لاحتياجات التطبيق.
- d. **سهولة الاستخدام والتعلم:** تتمتع Next.js ببنية متناسقة وواضحة، مما يجعل من السهل على المطورين فهمها والبدء في استخدامها. بالإضافة إلى ذلك، يوفر Next.js مجموعة كبيرة من الأدوات والمكتبات المساعدة التي تسهل عملية التطوير.
- e. **تكاملاً جيد مع React:** تعتمد Next.js على React كمكتبة أساسية، مما يعني أنها توفر تكاملاً ممتازاً مع جميع المكتبات والأدوات التي تعتمد على React . هذا يجعل من السهل إضافة المكونات وإدارة الحالة وتطبيق التأثيرات البصرية بشكل فعال.

### 3. TailwindCSS:

يعتمد المشروع على TailwindCSS لتصميم واجهة المستخدم بشكل سريع وفعال، مع إمكانية تخصيص التصميم بسهولة باستخدام فئات CSS المعرفة مسبقًا.

تم اختيار Tailwind CSS لمشروع TravelAid لعدة أسباب منها:

- سهولة الاستخدام والتخصيص:** توفر Tailwind CSS نهجًا فريدًا لتصميم الواجهات حيث يستخدم فئات CSS المعرفة مسبقًا لبناء المكونات. هذا النهج يجعل من السهل تخصيص التصميم وتطبيق التغييرات بسرعة، دون الحاجة إلى كتابة تعليمات CSS مخصصة.
- التنسيق الشامل:** توفر Tailwind CSS مجموعة كبيرة وشاملة من CSS-classes التي تغطي جميع جوانب التصميم، مما يسمح للمطورين بتطبيق أي تصميم بسهولة وفعالية.
- الأداء الجيد:** تتميز Tailwind CSS بأداء ممتاز، حيث تكون ملفات CSS الناتجة أكثر فعالية وصغر حجمًا مما يسهل تحميلها بسرعة وتحسين أداء الموقع.
- توفير الوقت:** بفضل المجموعة الواسعة من الفئات الجاهزة، يمكن للمطورين تصميم واجهة مستخدم متميزة دون الحاجة إلى كتابة CSS مخصصة من الصفر، مما يوفر الكثير من الوقت والجهد عند التطوير.

### 4. Prisma:

تم استخدام Prisma كأداة ORM للتفاعل مع قاعدة البيانات SQL Server بطريقة بسيطة ومنظمة، مما يسهل عمليات الاستعلام والتحديث، حيث توفر هذه الاداة بشكل عام ما يلي:

- سهولة الاستخدام والتعلم:** توفر Prisma واجهة سهلة الاستخدام للتفاعل مع قاعدة البيانات SQL، مما يجعل من السهل على المطورين فهم كيفية تنفيذ الاستعلامات وإدارة البيانات دون الحاجة إلى معرفة متعمقة في SQL.
- ORM متطورة:** توفر Prisma ORM (Object-Relational Mapping) طريقة بديهية للتفاعل مع قاعدة البيانات، حيث يتم تحويل الكائنات في التطبيق مباشرة إلى سجلات في قاعدة البيانات والعكس بكل سهولة.
- الأمان والاستقرار:** توفر Prisma ميزات أمان مدمجة مثل التحقق من الصحة (validation) والتحقق من الأمان (security checks) وتشفير البيانات (data encryption)، مما يساعد في حماية البيانات وضمان استقرار التطبيق.

- أداء ممتاز:** تتميز Prisma بأداء فائق، حيث يتم توليد الاستعلامات بشكل ذكي وفعال، مما يسمح بتنفيذ العمليات البيانات بسرعة وكفاءة.

- دعم متعدد لقواعد البيانات:** تدعم Prisma مجموعة متنوعة من قواعد البيانات SQL مثل MySQL و PostgreSQL و SQLite، مما يجعلها مرنة ومتوافقة مع مختلف البيئات التطويرية.

- تكامل مع تقنيات التطوير الحديثة:** يتكامل Prisma بسلاسة مع التقنيات الحديثة مثل GraphQL و Next.js و TypeScript، مما يسهل على المطورين بناء تطبيقات متطورة وقوية بسهولة.



## 5. SQL Server(2017):

تم استخدام قاعدة البيانات SQL Server لتخزين معلومات المستخدمين والرحلات والوجهات السياحية بشكل آمن ومنظم.

## 6. Cloudinary :

Cloudinary هي خدمة سحابية (Cloud Service) تقدم حلولاً لإدارة وتحسين الوسائط (الصور والفيديوهات) على الإنترنت. تهدف Cloudinary إلى توفير أدوات قوية ومرنة للمطورين والمصممين لتحميل وتخزين وتحرير وتسليم الوسائط عبر الويب بشكل مباشر وفعال.

تتميز Cloudinary بمجموعة واسعة من الميزات والخدمات التي تشمل:

تحميل وتخزين الوسائط: يمكن للمستخدمين تحميل الصور ومقاطع الفيديو والملفات الصوتية والملفات الأخرى على الخوادم السحابية لـ Cloudinary للتخزين والإدارة.

معالجة الوسائط والتحويل: توفر Cloudinary أدوات لمعالجة الوسائط، بما في ذلك تغيير الأحجام، وتحرير الصور، وتحويل الصيغ، وتطبيق التأثيرات، وإضافة النصوص، والمزيد.

تسليم الوسائط بسرعة: توفر Cloudinary شبكة توصيل محتوى (CDN) متطورة لتسريع تحميل الوسائط وتوفير تجربة مستخدم متميزة.

إدارة الوسائط بشكل شامل: يمكن للمطورين إدارة الوسائط بسهولة باستخدام لوحة التحكم الخاصة بـ Cloudinary، مما يتيح لهم تصفح وتنظيم وحذف الوسائط بشكل فعال.

أمان الوسائط: تقدم Cloudinary ميزات أمان متقدمة مثل التشفير والتحقق من الهوية وتحكم الوصول، لضمان سلامة الوسائط وحمايتها من الوصول غير المصرح به.

دعم تقنيات التطوير الحديثة: تتكامل Cloudinary بسهولة مع تقنيات التطوير الحديثة مثل ReactJS و Next.js و Vue.js و Angular وغيرها، مما يسهل على المطورين تضمين وإدارة الوسائط في تطبيقاتهم بشكل فعال.

باختصار، Cloudinary هي خدمة شاملة توفر أدوات متقدمة لإدارة وتحسين الوسائط عبر الويب، مما يجعلها الخيار المثالي للمطورين والمصممين الذين يبحثون عن حلول موثوقة وفعالة لإدارة الوسائط في تطبيقاتهم ومواقعهم.

### 1. React Icons :

لتوفير مجموعة كبيرة من الأيقونات الجاهزة للاستخدام في واجهة المستخدم، مما يعزز تجربة التصفح.

### 2. Leaflet :

يستخدم Leaflet لعرض الخرائط التفاعلية وتوفير تجربة استكشاف سهلة وممتعة للمستخدمين.

### 3. Zustand :

يوفر Zustand حلاً بسيطاً وفعالاً لإدارة حالة التطبيق في React، مما يسهل تبادل الحالة بين مكونات التطبيق بدون الحاجة إلى إضافة Redux.

### 4. React Hot Toast :

يستخدم React Hot Toast لعرض رسائل التنبيه بشكل جذاب وسلس، مما يحسن من تجربة المستخدم ويزيد من فاعلية التواصل.

### 5. Bcrypt :

يتم استخدام Bcrypt لتشفير كلمات المرور بشكل آمن، مما يحمي بيانات المستخدمين من الاختراقات والسرقة.

### 6. Axios :

يستخدم Axios لإجراء طلبات HTTP من العميل إلى الخادم بشكل سهل وفعال، مما يسهل التفاعل مع البيانات.

### 7. Date-fns :

يستخدم Date-fns لإدارة التواريخ والأوقات بشكل مرّن وفعال، مما يسهل عمليات التنسيق والتحويل.

### 8. React Date Range :

يستخدم React Date Range لتحديد نطاق التواريخ بسهولة وتفاعلية، مما يسهل على المستخدمين تحديد فترة الرحلة المفضلة.

### 9. Next-Clouinary :

يستخدم Next-Clouinary لتحميل وإدارة الصور بشكل فعال، مما يسهل عرض الصور والمعلومات المصاحبة للوجهات السياحية بشكل جذاب.

### 10. React Spinners :

يستخدم React Spinners لإضافة تأثيرات تحميل جذابة وسلسة، مما يحسن من تجربة المستخدم خلال فترات الانتظار.

## **.11 World-Countries :**

يستخدم World-Countries للحصول على معلومات حول البلدان والمناطق المختلفة، مما يسهل عرض الوجهات السياحية بشكل دقيق وشامل.

## **.12 React Hook Form :**

يستخدم React Hook Form لإدارة النماذج والإدخالات بشكل فعال ومرن، مما يسهل جمع بيانات المستخدمين والتفاعل معها بشكل سلس وسهل.

### 3- تفاصيل قاعدة البيانات:

تم انشاء قاعدة البيانات بواسطة Prisma مما يجعل العملية سهلة جدا وتلقائية بشكل كامل. إنشاء قاعدة بيانات باستخدام Prisma على SQL Server يتضمن عدة خطوات. هنا تفصيل للخطوات التي يمكن اتباعها:

#### 1. تثبيت Prisma CLI:

قم بتثبيت أداة Prisma CLI باستخدام الأمر التالي في موجه الأوامر:

```
npm install prisma -g
```

#### 2. إنشاء مشروع Prisma:

قم بإنشاء مشروع Prisma جديد باستخدام الأمر التالي:

```
prisma init
```

مع اتباع الخطوات لاختيار نوع قاعدة البيانات والإعدادات الأساسية لمشروع Prisma.

#### 3. تكوين ملف schema.prisma:

قم بفتح ملف schema.prisma الذي تم إنشاؤه في المشروع وقم بتحديد نموذج البيانات الخاص بك وتعريف الجداول والعلاقات.

#### 4. تكوين محرك SQL Server:

افتح ملف schema.prisma وقم بتكوين محرك SQL Server بتحديد النوع `provider` كـ `sqlserver` وتعيين الاتصال بقاعدة البيانات والمعلومات الأخرى المطلوبة للاتصال.

#### 5. توليد مخطط قاعدة البيانات:

بعد تكوين schema.prisma، قم بتوليد مخطط قاعدة البيانات باستخدام الأمر التالي:

```
prisma generate
```

سيقوم هذا الأمر بإنشاء ملفات TypeScript المرتبطة بنموذج بياناتك وسيولد مخطط قاعدة البيانات الخاص بك.

#### 6. تطبيق التغييرات على قاعدة البيانات:

إذا قمت بإجراء أي تغييرات على نموذج البيانات في schema.prisma، يجب عليك تطبيق هذه التغييرات على قاعدة البيانات باستخدام الأمر التالي:

```
prisma migrate dev --name <اسم التحديث>
```

ينشئ هذا الأمر تحديثات جديدة وينفذها على قاعدة البيانات.

```

generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "sqlserver"
  url      = env("DATABASE_URL")
}

model User {
  id          Int      @id @default(autoincrement())
  name        String?
  email       String?  @unique
  emailVerified DateTime?
  image       String?
  hashedPassword String?
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
  listings    Listing[]
  reservations Reservation[]

  @@map(name: "Users")
}

model Listing {
  id          Int      @id @default(autoincrement())
  title       String
  description  String
  imageSrc    String
  createdAt   DateTime @default(now())
  category    String
  roomCount   Int
  bathroomCount Int
  guestCount  Int
  locationValue String
  userId      Int
  price       Int

  user      User      @relation(fields: [userId], references: [id], onDelete: Cascade)
  reservations Reservation[]
}

model Reservation {
  id          Int      @id @default(autoincrement())
  userId      Int
  listingId   Int
  startDate   DateTime

```

```

endDate      DateTime
totalPrice    Int
createdAt     DateTime @default(now())

user          User      @relation(fields: [userId], references: [id], onDelete: NoAction,
onUpdate:NoAction)
listing       Listing    @relation(fields: [listingId], references: [id], onDelete:
Cascade)

@@map(name: "Reservations")
}

```

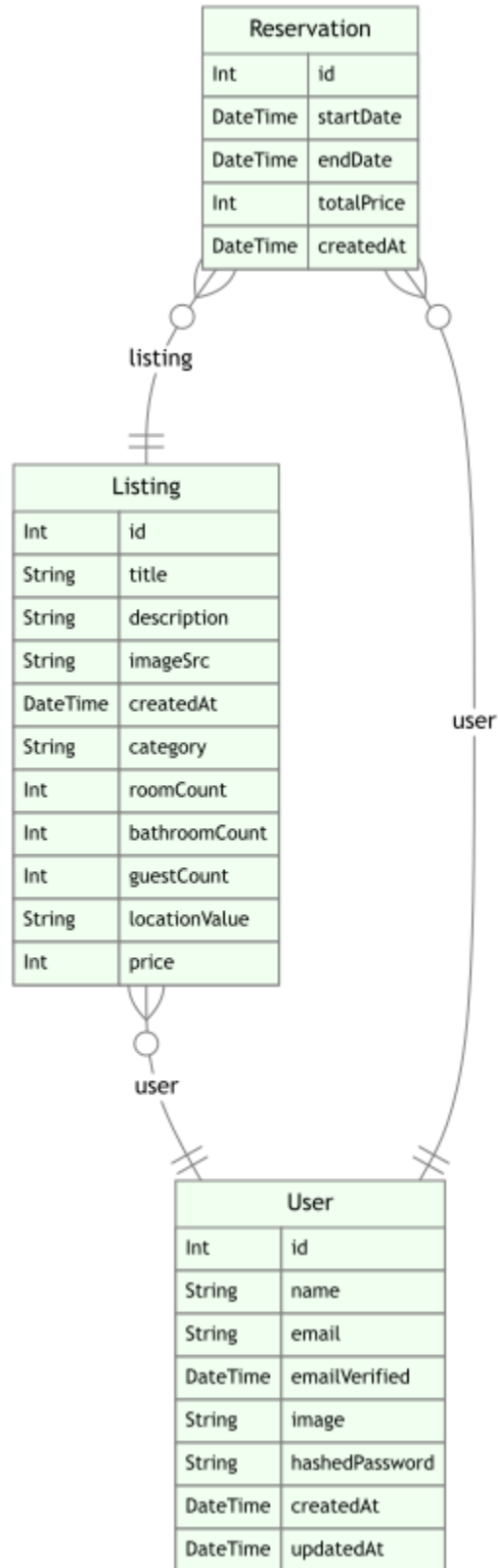
حيث:

generator client: تعريف مولد يسمح بإنشاء كود للواجهة البرمجية لقاعدة البيانات باستخدام Prisma Client للاتصال بقاعدة البيانات.

datasource db: تعريف مصدر البيانات الذي يحدد نوع مزود قاعدة البيانات (في هذه الحالة sqlserver) وعنوان URL لقاعدة البيانات المستخدمة، والذي يستخدم قيمة المتغير DATABASE\_URL الموجودة في ملف (.env) وتستخدم التعليمة @@map لتحديد اسم الجدول في قاعدة البيانات لكل نموذج، مما يوفر تحكمًا دقيقًا في الاسماء المستخدمة في قاعدة البيانات.



## المخطط العلائقي لقاعدة البيانات:



## 4- تشغيل المشروع محلياً:

ملاحظة: بيئة العمل التي تم استخدامها في بناء المشروع هي: **VisualStudioCode**

1. تثبيت الأدوات الأساسية:

قم بتثبيت Node.js و npm إذا لم يكنا مثبتين بالفعل على جهاز الكمبيوتر الخاص بك.

2. استعادة قاعدة البيانات:

- قم بفتح Microsoft SQL Server Management Studio.
- انقر بزر الماوس الأيمن على "قواعد البيانات" واختر "استعادة قاعدة بيانات".
- حدد ملف النسخ الاحتياطي travelaid.bak الذي تم إرفاقه مع ملفات المشروع ضمن مجلد (Backup) واستعد قاعدة البيانات منه.
- انتظر حتى تكتمل عملية الاستعادة.

3. تكوين ملفات البيئة:

- قم باستبدال ملف env.example بملف (.env) الموجود ضمن ملفات المشروع.

4. تثبيت الاعتماديات:

- افتح موجه الأوامر في مجلد المشروع، أو ضمن نافذة Terminal داخل محرر الأكواد **VisualStudioCode**.
- قم بتشغيل الأمر التالي لتثبيت جميع الاعتماديات (dependencies):

```
npm install
```

5. تطبيق التحديثات لقاعدة البيانات:

- استخدم Prisma CLI لتطبيق التحديثات اللازمة لقاعدة البيانات باستخدام الأمر:

```
npx prisma migrate dev
```

6. توليد ملفات المخططات:

- قم بتوليد ملفات المخططات باستخدام الأمر:

```
npx prisma generate
```

7. تشغيل الخادم المحلي:

- بعد إكمال الخطوات السابقة بنجاح، قم بتشغيل الخادم المحلي باستخدام الأمر:

```
npm run dev
```

8. فحص التطبيق:

- افتح المتصفح وانتقل إلى عنوان <http://localhost:3000> لفحص تشغيل التطبيق.
- بعد اتباع هذه الخطوات، يجب أن يكون التطبيق جاهزًا للاستخدام ويمكنك البدء في اختبار الميزات .

1. **ReactJS:**

- Documentation: <https://react.dev/learn>
- GitHub Repository: <https://github.com/facebook/react>

2. **Next.js:**

- Documentation: <https://nextjs.org/docs/getting-started>
- GitHub Repository: <https://github.com/vercel/next.js>

3. **Tailwind CSS:**

- Documentation: <https://tailwindcss.com/docs>
- GitHub Repository: <https://github.com/tailwindlabs/tailwindcss>

4. **Prisma:**

- Documentation: <https://www.prisma.io/docs/>
- GitHub Repository: <https://github.com/prisma/prisma>

5. **SQL Server:**

- Documentation: <https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-2017>
- Download and Resources: <https://www.microsoft.com/en-us/sql-server>

6. **Cloudinary:**

- Documentation: <https://cloudinary.com/documentation>

7. **Next Cloudinary:**

- GitHub Repository: <https://github.com/cloudinary/nextjs>