universidade de aveiro
departamento de electrónica,
telecomunicações e informática

Professional and Social Aspects

of Informatics Engineering

Work no. 4
PropertEase - A Unified Property Listing Manager

Bárbara Galiza
NMec: 105937

Diana Miranda
NMec: 107457

João Dourado
NMEC: 108636

Miguel Figueiredo
NMEC: 108287

Ricardo Quintaneiro
NMEC: 110056

Rúben Garrido
NMec: 107927

June 8, 2024

Contents

1 Introduction

PropertEase is an essencial application that establishes a unified and intelligent control panel aimed at simplifying the management of property listings for owners. The key features of the application are:

- **Global property management**: PropertEase aggregates all the properties that are associated with the user's login across all available listing services. This provides the user the possibility to manage all their properties across all listing services from this application alone, rather than repeat the same management actions individually, on each service.
- **Calendar Synchronization**: This feature ensures that when a reservation is made on one of the listing services, PropertEase picks up this event and sends the update to the others listing services where this property is also listed, thus blocking those dates. In this way, booking conflicts on the same dates are automatically prevented, making it easier for both the customer and the property owner.
- **Dynamic price recommendation**: Dynamic price recommendation feature is of key importance in PropertEase, enabling property owners to use a strategic rental pricing with the aim of maximizing revenues. This feature provides for making competitive pricing decisions aligned with market expectations, based on the analysis of external demand indicators and comparison with similar properties. It also uses ML techniques to help predicting the best prices. See Figure 1 for more information about the recommendation process.
- **Data collection**: The data collected from all these system instances will enable stakeholders (municipalities, Destination Management Organizations and land developers) to make more informed decisions.

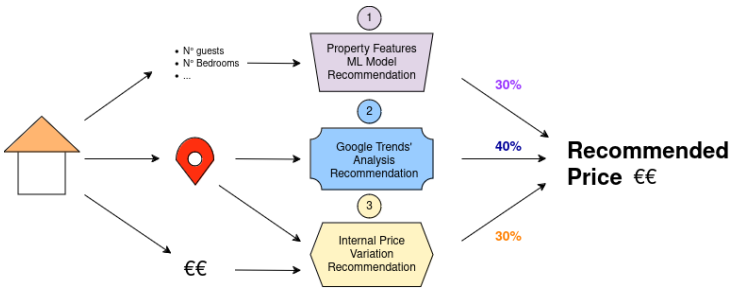

Figure 1: PropertEase's price recommendation process

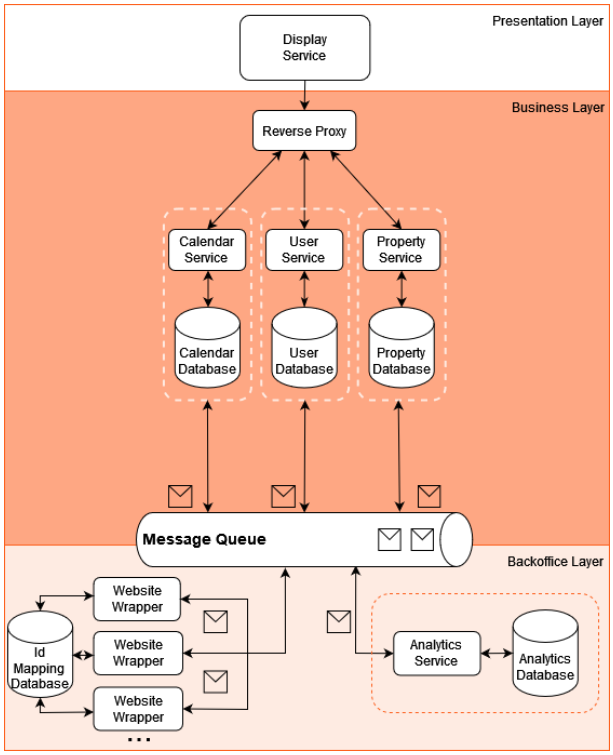

Figure 2: PropertEase's Architecture

The system complies with a microservices architecture, prioritizing modularity. It is divided in 3 main layers: Presentation, Business and Backoffice (Fig. 2).

The Presentation Layer, built with React, provides an easy-to-use user experience, allowing access to all PropertEase features. This layer communicates through a Reverse Proxy with the Business Layer that contains multiple services:

- User Service - service responsible for user management and external service connection logic;
- Property Service - centralized service of properties from multiple platforms via a universal schema;
- Calendar Service - centralized service of all reservations and property maintenance events.

The Business layer communicates asynchronously with the Backoffice Layer that integrates with external property listing platforms through website wrappers to detect new properties or reservations and to propagate property detail or availability updates. The Analytics Service, also part of the Backoffice layer, handles operations related to the recommendation price algorithm and stores relevant data and statistics to the stakeholders in an Elasticsearch database.

2 Ecosystem Positioning

2.1 Potential suppliers for PropertEase

When it comes to physical equipment, PropertEase does not have any dependencies, as it is a digital service for property management, therefore, there would not be a requirement for a physical equipment supplier.

On the other hand, we would require the use of a cloud web-hosting platform, such as *AWS (Amazon Web Services)* or *GCP (Google Cloud Platform)* to host our different microservices - this will be explored deeper in the hyperscalers section.

Another requirement is protection of attacks such as *DDoS (Distributed Denial of Service)* attacks. PropertEase being hosted on a pay-as-you-go cloud platform, means that being vulnerable to *DDoS* attacks is high risk, not only because of damage done when it comes to customer trust, but also raw operation costs caused by these attacks in these cloud platforms. A possible provider for *DDoS* protection is *Cloudflare*.

Another key supplier for PropertEase are property listing services, for which we would require API permission to operate efficiently. The more listing services our system supports, the greater the value to end customers, especially if we support the biggest shareholders in the online travel agency market.

2.2 Potential customers and their profiles

The profile of the main end-user of PropertEase are exclusively property owners that have to deal with managing their properties across multiple listing services at the same time. The scale at which they manage properties is variable, we could have clients with a single property, but also clients with dozens of properties to manage. They will have their properties listed on multiple listing services - definitely listed on the biggest players of the market (*Airbnb*, *Booking.com* and *Expedia*), and possibly in some nationally relevant smaller listing services.

Our clients deal with problems such as overbooking, paying attention to new reservations and manually updating calendars across all listing services, and updating prices manually in each service whenever they want to change them globally.

All our customers will most likely be using most of *PropertEase's* features, and, for that reason, usability tests with users who fit this profile, ideally with data similar to what they work

with, would be extremely valuable to the delivery of a quality product.

2.3 Competitors and differentiating factors of PropertEase

There are multiple competitors to PropertEase we have identified, namely, *Avantio*[1], *Avaibook*[2], *Holidu*[3] and *Icnea*[4].

Some features are common among all competitors and PropertEase alike, such as: calendar synchronization among different listing services; price (and other property details) synchronization among different listing services.

One feature that makes PropertEase stand off, is our **dynamic pricing algorithm** using machine learning prediction combined with *Google Trends* data to attribute properties prices. The only competitor above that offers a similar feature is *Icnea*, but, unlike PropertEase, offers it as an external integration [1] with businesses dedicated to property pricing, and doesn't mention changing prices automatically, as PropertEase allows.

Another feature that is exclusive to our system is the possibility of creating **management** events for a property, such as cleaning or maintenance events. These events close out the property from receiving reservations for the given time interval.

2.4 Regulatory framework associated with PropertEase

PropertEase, being a service that aggregates multiple listing service, requiring use of their APIs - APIs that do not belong to us, nor are controlled by us - avoids some difficulties related to regulation when it comes to proving ownership of properties or proof of being of legal age to make a reservation.

For example, *Airbnb*[5] requires users to provide proof of property ownership when registering a property for booking and proof of adulthood when making reservations.

Although supported listing services ensure compliance with these legal requirements, we still fall under the *General Data Protection Regulation (GDPR)* by operating inside the European Union. This means we should protect and ensure privacy of the data of our clients, such as their properties and reservations from their clients.

[1] https://www.avantio.com/
[2] https://www.avaibook.com/en/
[3] https://www.holidu.com/partners
[4] https://icnea.us/property-management-system
[5] https://www.airbnb.com/

Another important piece of regulation to abide by is the European AI Act, to guarantee the use of AI is safe, ethical and respects human rights. This will be discussed in more detail in section 11, Ethical Considerations.

2.5 Business Scalability

Our product, being a property management system that relies on listing services being available to share their users' data to be managed by us, is highly dependent on these services.

The biggest limiting factor for our scalability would be what listing services we support in PropertEase. Ideally, we would support all the "big players" in the *Online Travel Agency (OTA)*[6] market, such as *Booking.com*, *Airbnb* and *Expedia*. If this is the case, we could attract many property owners to use our system, as they are very likely to use, and have most their revenue, coming from these listing services.

[6]https://www.statista.com/statistics/1039616/leading-online-travel-companies-by-market-cap/

3 Monetization Strategy

In this section, we outline three different strategies for monetization, and, at the end, choose the one we think makes the most sense for being a long term profitable and thriving business.

3.1 Investigation Project

PropertEase could be considered an investigation project for the *ATT - Acelerar e Transformar o Turismo* project, being funded directly by the *Plano de Recuperação e Resiliência (PRR)* fund. In this context, all the system's features would be totally free, as the main source of value would be all the collected data, which would aid other investigation projects within the *ATT* project, or even sales to third party tourism organizations.

Advantages

- **Consistent and stable financing**: Project developers would be paid in a predictable and consistent manner, not worrying about how successful our business is, within the property management system market, as a source of monetization;
- **Reduced risk of financing**: Instead of worrying about financing early on - an issue commonly found in projects, as they are basically a source of money loss until an MVP is delivered, we can focus on the project without pressure for commercial returns;
- **Easier to scale number of users**: As a completely free investigation project focused on obtaining tourism-related data, the absence of costs would encourage widespread use and word-of-mouth promotion.

Disadvantages

- **Limited time funding**: An investigation project has funding as long as it exists. After the project effectively ends, funding is cut, and the project becomes an asset of whatever institution is funding it, so developers of the project would not receive long term funding;
- **Data selling**: As an investigation project, we do not own the data our system has, our financing institution does, so selling data to third parties would not be of our responsibility and we could not profit off of it.

3.2 Freemium Monetization Model

Another approach could be offering *PropertEase* in a *freemium* model. We could offer most features such as automatic calendar and price synchronization across multiple services freely, but lock other features behind a **subscription-based** paywall, such as: sending keycodes for opening the door to a property to a client; and our dynamic pricing algorithm based on market trends.

With this *freemium* model, we would offer the possibility of a 1 month free trial of the **Premium** tier for all users, which might incentivize some users to start using our service, while also paying for it in a **subscription** model.

As for free users, to make up for our operation costs, they would see ads throughout the system.

Advantages

- **Lower barrier of entry**: Being able to freely use most features of our system means more property owners will be incentivized to try the system out, as most property management systems are locked behind some kind of paywall.
- **Data selling**: Free tiers incentivize more users to try out our system, and that would mean more data gets collected. increasing the volume of data collected. This data, compliant with GDPR requirements, could potentially be sold to interested third parties.
- **Ads subsidizing the cost of free features**: For free users, ads could be placed around the app to make up for the cost of operation and even generate profit.
- **Per-property price premium tier**: The cost of a subscription to the premium tier could scale off of how many properties a user owns. This is a great advantage compared to competitors that offer a static price subscription.

Disadvantages

- **Limited features in free tier**: The fact that the free tier has very limited features might drive users away instead of incentivizing them to switch to a premium tier.

3.3 SaaS - Software as a Service

Another potential approach for monetization would be to sell parts (for example, specific microservices) of *PropertEase* as an individual module to be integrated in other businesses. Our dynamic pricing algorithm that uses real market trends for attributing price recommendations would be a great addition to any thriving business that handles real estate pricing.

Another example would be to sell our data aggregation pipeline, that gets data from external APIs and aggregates that data in a microservices architecture, to companies in completely different markets - for example, a sports data aggregation platform, that takes in information from multiple sports APIs and displays them to their users.

Advantages

- **Market independence**: We could sell certain modules of our system to businesses in different markets, such as a sports data business as mentioned previously, and would not depend on the tourism market to thrive, which, as we saw in the *COVID-19* pandemic, is not a totally safe market.
- **High cost of switching for users**: If users decide to integrate a certain module of ours into their business, the cost of switching out to a new module is very high, as many changes might be required to make the transition. This reduces churning rate, as users will be more locked into our module

Disadvantages

- **Harder to reach more users**: It would be way harder to reach more users by, for example, word-of-mouth spread, since users are now businesses looking to use one of our modules in their system, and it would be harder to "spread the word" of the existence of these modules, especially if these modules are profitable - if it makes them profit, they probably won't tell other similar businesses
- **More investment in marketing**: As said previously, it would be difficult to benefit from networking effects. To reach more users, we would need a significant investment in marketing, instead of exploiting natural network effects as other monetization strategies, such as *freemium* might.

3.4 What we choose

Being an *Investigation Project* as an approach for funding/monetization is **not a desirable choice** for our project, as the disadvantages greatly outweigh the advantages. Ideally, we should be independent of outside funding and would not depend on limited time funding, which is the case for most investigation projects, especially publicly funded ones.

On the other hand, *freemium* is a great choice for our business, as, if the *free* and *premium* tiers are well defined and incentivize users to transition from the free to the paid version due to the greater benefits such as the dynamic pricing algorithm, then that combined with the exploiting of networking effects benefits make it a the ideal choice.

As for the option of being a *Software as a Service* and selling parts of our system, it is great if we can get a large enough client base, as they might be "locked" into using our software, since the cost of switching out might be high. On the other hand, the marketing investment to get our product to reach possible clients makes it less valuable than the *freemium* model.

The *freemium* monetization strategy is the one that adapts the best to our business context, being a property management system with many features, that could offer a portion of those **freely** and some specific features for **premium** users.

4 Legal Limitations and Viability

A project of this scale, involving integrations with platforms like Booking, Airbnb, and others, carries with it some legal limitations.

Operating mostly within Europe, PropertEase needs to worry about compliance with the GDPR. However, since the data used by PropertEase comes in from APIs provided by the listing platforms it connects to, PropertEase needs to take care about regulations and policies from those, too.

Regarding user data and its safety, PropertEase will only have access to the data provided by listing platforms through their APIs, which have complied with GDPR and all security measures. However, PropertEase shall ensure that data transfers between PropertEase and the platforms respect data protection regulations, such as GDPR, and that users personal data is well handled and international data transfers respect the mechanisms for GDPR adequacy. However, this will be further explored in the section 9.

For us to be able to sell data, which is one of the possible business models, it is fundamental that we do it in a fair and legal way. Firstly, we need to get the users explicit consent to collect their data and commercialize it. Moreover, any information relating to personal or sensitive data needs to be fully anonymized so that privacy and protection are possible. Only by following these strict procedures are we able to contemplate this business possibility in an ethical way, respecting current regulations on data protection, such as GDPR.

By considering all those aspects, PropertEase seems to fulfill all the legal standards and conditions while it also complies with the most important regulations and respects users privacy.

5 Intellectual Property

For PropertEase, intellectual property is a subject of core importance. However, the fact that there are already a few property management platforms available in the market, which are pretty similar to PropertEase, means that it is not something that can be patented since it is not an innovation or a creation. Although, the visual identity, the brand, and the name PropertEase can be protected.

It is very important to register the brand and name PropertEase because, apart from giving more visibility, it provides more credibility to the users since a trademark is associated with professionalism and it can impact the decision of users on whether or not to make use of the product. It is also a means of differentiating the product within the market, which provides a symbol of assurance, which can increase the value of the firm.

This trademark ensures a legal monopoly, an exclusive right to the brand, and the right to prevent third parties from using the product without authorization.

Proprietary access to APIs from hosting platforms and services is another necessary aspect of the intellectual property of PropertEase. The proprietary access states that PropertEase is allowed to use, through licensing agreements, the said APIs. Those licensing agreements are part of the intellectual property strategy in that they offer legal protection of the ability to use specific technologies and services that build out the functionalities of the site. That is, PropertEase is allowed to integrate certain features like calendar synchronization and property detail updates in a unique and very effective way.

6 Open Source Software

All open-source software used in this project is licensed for commercial usage. The table below shows each software's license:

Technology	License
React	MIT License [2]
Vite	MIT License [3]
TypeScript	Apache License 2.0 [4]
Flatpickr	MIT License [5]
DaisyUI	MIT License [7]
Nginx	2-clause BSD License [8]
FastAPI	MIT License [9]
RabbitMQ	Mozilla Public License v2.0 and Apache License 2.0 [10]
MongoDB	Server Side Public License v1.0 and Apache License v2.0 [11]
PostgreSQL	MIT licenses [12]
SQLite	Public Domain [13]
Python	Python Software Foundation License Agreement [14]
SerpAPI	MIT License [15]
Elasticsearch, Logstash & Kibana	Elastic License [16]
Inside Airbnb (Dataset)	Creative Commons Attribution 4.0 International License [17]
ML Model (Kaggle)	Apache License 2.0 [18]
Firebase	Apache License 2.0 [19]
React Auth Kit	MIT License [20]

Table 1: Licenses for technologies used in the project

6.1 Copyleft Licenses

RabbitMQ is licensed under the Mozilla Public License 2.0, which means that all code where it is used must be open sourced as well. However, since this project is closed-source, a new viable alternative must be found, such as Kafka, whose license is Apache License 2.0.

MongoDB Server is licensed under the Server Side Public License 1.0, a copyleft license forked from the GNU Affero General Public License 3.0. Like on the Mozilla Public License, this means that code should be released to the public. Either an alternative must be found

if using PropertEase as a public product, or a commercial license must be purchased from MongoDB.

6.2 Permissive Licenses

The MIT License and the Apache License 2.0 are permissive enough and don't bring any problems to our software context. Moreover, Apache License 2.0 grants explicit patent rights from contributors to users, protecting against patent litigation.

The 2-clause BSD License is permissive as well, and very similar to the MIT license, where a copyright notice must be included in all copies.

Both the Python Software Foundation License and the Elastic License, although non-standard, allow for usage in commercial and closed-source projects, which grant permissions to PropertEase to freely use products licensed by them.

The Creative Commons Creative Commons Attribution 4.0 International License present in the dataset allows sharing and adapting it, as long as the license terms are respected: attribution is mandatory and no additional restrictions can be applied over the data.

6.3 Public Domain

SQLite is in the Public Domain, therefore, a license is not required and this project can use it freely, with no limitations.

7 Cybersecurity

7.1 Major attacks PropertEase could face

Like any complex and interconnected system, our property management platform is susceptible to numerous threats. However, given its microservices architecture and the reliance on external services, it means it has some unique vulnerabilities. As such, the primary types of cyberattacks that PropertEase could face are:

- **Credential stuffing**

  Consists of using leaked credentials from data breaches of other websites or services to gain unauthorized access to user accounts. This type of attack exploits the common practice of password reuse, where users use the same password across multiple sites.

  Attackers can perform malicious activities such as stealing sensitive information, making unauthorized transactions, or using the account as a stepping stone for further attacks within the system.

  Because we use OAuth 2.0 as an authentication method with Google accounts, if an attacker has access to the Google account then it has access to the user account in our system.

- **Distributed Denial of Service (DDoS)**

  These are coordinated Denial of Service attacks from multiple sources to overwhelm the system through massive amounts of *fake* traffic, nowadays mostly done through the use of a botnet, making it unavailable to legitimate users.

  Prolonged downtime can lead to financial losses, not only due to the costs associated with mitigating the attack and restoring normal service, but because it can damage the reputation of our platform, driving users away and therefore losing potential income.

  As our system is made up of microservices, the whole system is more secure against these types of threats but can still be affected as a whole.

- **Data Breaches and Data Exfiltration**

  It happens as unauthorized access to sensitive user data, by means of external threats that gained access to a user account or by insider ones in the form of people that have access to restricted information, leading to data theft and privacy violations.

  Data breaches and exfiltration harm individuals' trust and confidence in PropertEase as well as our organization may face significant financial losses due to fines, legal fees, and the cost of remediation efforts.

  Even though PropertEase generates little information about it's users, it has access to a lot of sensitive user and property data that comes from the external listing platforms, as well as the calenderization of property events.

- **External Services APIs Exploitation**

  The abuse or manipulation of APIs provided by the integration of the external listing services to carry out unauthorized actions from our system or gain access to restricted data. This could include techniques such as excessive API calls, injection attacks, or parameter tampering to bypass security measures and exploit vulnerabilities in the external service interfaces.

  Unauthorized actions performed through manipulated APIs can compromise the integrity of data, resulting in incorrect or malicious data being introduced into our system and the systems of the external listing websites.

  Exploitation of external APIs can damage our organization's reputation due to the perceived lack of security, as well as being subject of legal actions resulting in financial losses.

- **Supply Chain attacks**

  Involves compromising third-party components or services (e.g., external APIs, libraries) integrated into our system to introduce vulnerabilities. These attacks exploit the trust relationships between our platform and external suppliers in the form of routine updates or new implementations, making the infiltration of our infrastructure possible.

  Once the malicious components are within our system, attackers can exploit the vulnerabilities to gain unauthorized access, steal data, disrupt operations, or further compromise other parts of the platform.

Our cybersecurity measures must be robust enough to prevent such attacks and guarantee the continuous operation of our property management platform. This means implementing an approach that not only defends against external threats but also monitors for any internal anomalies. We need to be vigilant about securing every aspect of our system, from the code we write to the third-party components we integrate.

7.2 Cybersecurity legislation PropertEase needs to comply with

Operating our property management system in the European Union, means our platform must comply with several key pieces of legislation and regulations related to cybersecurity. Here are the main ones to consider, with special focus on Portugal legislation, as well as the descriptions of the major subjects pertaining to those laws:

- General Data Protection Regulation (GDPR) and *Lei da Proteção de Dados Pessoais*

  Data Protection by Design and Default – Implement technical and organizational measures to ensure data protection principles are incorporated into the processes.

  Data Subject Rights – Ensure mechanisms are in place to allow data subjects to exercise their ARCO rights, namely access, rectification, cancellation (right to be forgotten) and objection to data processing.

  Data Breach Notification – Report data breaches to the relevant supervisory authority (CNPD - Comissão Nacional de Proteção de Dados) within 72 hours of becoming aware of the breach and, in some cases, notify the affected data subjects.

- E-Privacy Directive

  Confidentiality of Communications – Ensure the confidentiality of communications by protecting data in transit and requiring consent for interception or surveillance.

  Cookies and Tracking – Obtain user consent before placing cookies or similar tracking technologies on their devices, with clear and comprehensive information provided about their purpose.

- Cybersecurity Act

  Certification – Voluntary certification of the cybersecurity processes in our system, providing assurance to users and services that work with us about the reliability and robustness of our platform.

7.3 Detailed implementation of cybersecurity solutions

It is by staying ahead of potential threats that we can protect our platform and provide our users with a reliable, secure service, safeguarding trust and our companys reputation. To achieve this, the implementation of robust cybersecurity solutions demands an approach that integrates multiple layers of security minded practices and technologies. Here are the principal strategies that are addressing various aspects of security in our platform:

- Static Code Analysis and CVE/CWE Alerts
  - We identify and mitigate vulnerabilities in the codebase during development through the use of tools like SonarCube in our CI/CD pipeline that allow not only static code review from the point of view of maintainability but also Static Application Security Testing that scans for Common Vulnerabilities and Exposures - CVE and Common Weakness Enumeration - CWE.
- Intrusion detection and prevention
  - Our system deploys an Intrusion Detection System (IDS) and a Intrusion Prevention System (IPS) to monitor and respond to unusual activities that might indicate zero-day attacks or unauthorized access and malicious activity.

- Monitoring of atypical changes and logging
  - We implement a machine learning-based anomaly detection system that detects and respond to unusual activities, such as unexpected major price changes across multiple properties, and alerts the administrators.
  - We use tools like ELK Stack (Elasticsearch, Logstash, Kibana) to aggregate and analyze logs for unusual patterns.
- Administrative timing restrictions and control
  - Established rate limiting on critical administrative actions, for example, an admin can only delete a certain number of users within a specified period.
  - The system uses Role-Based Access Control (RBAC) to ensure administrators have only the necessary permissions for their roles.
  - We maintain detailed audit logs of all administrative actions and regularly review these logs for any unusual activities.
  - Multi-Factor Authentication (MFA) is required for all administrative access to add an extra layer of security.
- Data storage
  - PropertEase stores the minimal amount of data necessary for the functionality of the system, and this data is distributed/divided across its microservices which allows the segregation across the different databases. As such this data can be considered pseudonymized, which makes an attack on user and property data more difficult due to the need of correlation between these data points.

Implementing these cybersecurity solutions makes our system more secure and will help protect PropertEase against a wide range of threats. By integrating static code reviews, monitoring for zero-day vulnerabilities, detecting atypical changes, enforcing administrative controls and ensuring data segregation, we can have a safer system that safeguards user data and maintains trust with our clients.

8 Recovery Process

A robust recovery process is essential for ensuring that our property management system can quickly and effectively recover from disruptions, whether caused by cyberattacks, hardware failures or other incidents. Given our use of a microservices architecture through the use of Docker containers, we have some inherent advantages, such as isolated points of failure and scalable services.

The defining of a Disaster Recovery Plan (DRP) would be essential, and we would start by identifying potential risks and their impact on the system, setting recovery objectives for each microservice, ensure we have regular testing practices and establish recovery documentation. This would also have the processes to report attacks to CNPD and establish transparency to our clients.

Data backup and restoration would obviously have a major impact on weather we could recover efficiently from data loss, and this means scheduling regular backups of databases (Postgres, MongoDB) using automated tools, distributing the storage geographically to prevent data loss from regional incidents and regularly verifying the integrity of all backed-up data.

Because our system uses NGINX for distribution of traffic across the microservices we offer better service resilience that can be complemented with load balancing across deployed instances. We also have health checks that monitor the status of the microservices and automatically reroute traffic if a service instance fails. If a microservice is experiencing high-load and needs to scale, our system in deployment can use AWS Auto Scaling to automatically scale services based on demand.

As mentioned before, incident monitoring and response is very important and our system uses the ELK Stack to monitor logs and has a monitoring and alert system that ensures that unusual activity doesn't fly under the radar. We have a incident response plan set up outlining roles, responsibilities, and procedures for handling incidents, as well as post-incident reviews to identify root causes and improve the recovery process.

9 Privacy Protection

PropertEase gets all data from external APIs: user data is provided by Google's OAuth 2.0 authentication, whereas property data is pulled from external booking services, like Airbnb, Booking.com, and more.

Since those external services operate in the European Union, it is implied that their data already complies with the GDPR, thus exonerating this project from privacy protection responsibilities regarding the data collection.

By acting as a mediator between several services, PropertEase must ensure that both the connection to those APIs and the data storage act in accordance to the GDPR. This means that we must make sure data doesn't get leaked and only relevant data is fetched and stored from those services.

Concerning the AI model (more on that in section 10), we adapted it so that training data has no personal information or any kind of details that identify a property or a user. Therefore, GDPR doesn't apply to this model [27], since all data is anonymous, as per Article 26.

9.1 Illegal access & privacy issues

Despite only saving data from external services (which already comply with regulations), some security and privacy concerns arise regarding fetching APIs. In the case where the system's security is compromised, hackers can freely interact with APIs from all external services, which imposes a privacy risk: data can be deeply fetched and even modified in upstream servers. Following this, not only our system is compromised, but also all connected services to PropertEase, which can cause a severe data leak, with possible damages to all related companies.

This way of hacking can go even further, with booking companies secretly trying to hack PropertEase to fight competition among reservation platforms.

9.2 Preventing the risk

While not 100% effective, some security measures can prevent this from happening, such as:

- **Detect quick, bulk price changes**: if several properties of different owners have their price changed in an odd way and in a short matter of time, then the system can understand that a hack may be ongoing and thus take actions, automatically or manually, by notifying the system administrators.

- **Rate limit**: each user can only perform a maximum number of operations per minute, so that bulk changes cannot be made at a single time.

10 Artificial Intelligence

As previously mentioned, PropertEase provides a price recommendation feature, which uses a machine learning model in one of it's components. The model used is Random Forest Regression, which is part of the supervised learning category, and it is used to predict the price of a property based on it's features. The features being used are: number of guests (allowed per stay), number of bedrooms, number of beds, number of bathrooms, number of amenities, latitude and longitude. The code for the implementation of this model was based on a public, open-source notebook on Kaggle [21].

The data used to retrieve these features was obtained from the Inside Airbnb project, which is a mission driven project that provides data and advocacy about Airbnb's impact on residential communities [22]. It openly provides data from several cities across the globe, a big portion of them being European. Giving this is public data with a Creative Commons Attribution 4.0 International License, there aren't great constraints as long as it stays available. In case of a shutdown of this source, we could use PropertEase's own properties to train the model, once the number of properties tracked in the system can be considered big enough for training.

Regarding the potential issues tied to the data used for training, there are primarily two:

1. The bias that may arise from the training only using some of all the cities where the actual system's properties are located

2. The latitude and longitude features may jeopardize potential high quality properties based on it's location

For mitigating the first, tests have been done and showed that, even when training with a city (e.g. Lisbon) and testing with a different one in the same country (e.g. Aveiro), the performance remains similar and the predictions remains reasonable. Still, to reduce the bias even more, the best solution would be acquiring a more complete dataset, what could be achieved, as previously mentioned, trough the usage of PropertEase's own properties.

As for the second, when comparing two properties with different coordinates values but with the same values on the other features, it was shown that difference in the price prediction was small. Additionally, as much as the usage of coordinates might put good properties in disadvantage, the algorithm simulates a behavior that is already present in society, which is determining the value of a property based on it's location and surroundings.

Moreover, both of these issues are attenuated by the fact that the predictions are not directly used as the recommended price, but instead used to obtain the median price, which is then compared to each property based on each feature, excluding the coordinates. Furthermore, the AI recommended price is just one of three components used to calculate the final recommended price, and the other two do not use any type of artificial intelligence.

Another important aspect of the countering of bias is the security in learning systems, since an attacker could cause the model to produce incorrect biased results. To achieve that, the security measures addressed in chapter 7 and 8 are crucial for this part of the system, specially the constant monitoring. In addition, the usage of an open source model and an openly available dataset allows the verification against potential implicit bias by anyone.

Finally, both issues are also fully disclosed to the users, so that they can take an informed choice on whether to apply or not our recommended price.

11 Ethical Considerations

As in most informatics systems, there are ethical aspects to be considered regarding ProperEase. There are mainly two: the potential rise of prices for property guests and the potential loss of money for property owners, both due to the price recommendation feature.

When it comes to the property guests, the price recommendation might impose higher costs due to the market trends analysis and due to the AI predictor, presented in the last section. Regarding the market analysis, which is done with Google Trends data, while it might be seen as unethical or unfair by some, the feature on itself is not the source of the problem, but a facilitator of a practice that has already been done by property owners even before technology. It's natural that a property owner would rise prices knowing a big event will occur in their city.

Regarding the AI predictor, it is the property owner responsibility to apply or not the recommended price, knowing it may impose higher costs to their clients. That also applies for the market trends analysis. To make this issue clear to the property owner, we include a disclaimer in the Terms and Conditions that explains that this feature might negatively affect the buyers, and that all responsibility of using the recommended price is of the property owner.

In the other hand, the price recommendation feature might cause the property owner to loose money, since it is probabilistic and therefore, not perfect. For this situation, we also include in the previously mentioned Terms and Conditions document, the disclaimer that the user might loose money due to the probabilistic nature of the algorithm, and that they may use it at their own risk.

To conclude, in a ethical standpoint, the EU AI Act legislation has to be considered. In that sense, the AI component of the price recommendation feature would be categorized as Limited Risk or as Minimal Risk, according to [25] and [26]. That implies the following of transparency obligations such as "inform any person exposed to the system in a timely, clear manner when interacting with an AI system" and "Where appropriate and relevant include information on which functions are AI enabled, if there is human oversight, who is responsible for decision-making, and what the rights to object and seek redress are" [25]. It also obligates the constant monitoring and improvement of the model, which would also be done.

12 Relationship with Hyperscalers

As mentioned in Section 3, we have chosen the Freemium model as our business strategy. Therefore, it is important to consider the pros and cons of using hyperscalers and to keep them aligned with our strategic and operational goals. Adopting a no-hyperscalers strategy grants more control over the resources but limits scalability and may increases the operational complexity. On the other and, by choosing a with-hyperscalers strategy, it is possible to have infinite scalability and high availability but with variable and unpredictable costs.

Given the microservices architecture of PropertEase, which is highly modular, the transition from no-hyperscalers to with-hyperscalers strategies would not require extensive changes in the architecture and would not be time-consuming or technically difficult. So, for a safer approach, initially we would opt for a strategy without hyperscalers, implement a self-hosting system capable of handling a large number of users to assess whether people would really like and use PropertEase. And in case the growth of users becomes exponential, then we would move over to using hyperscalers so as to handle much larger scales and unexpected peaks.

To use hyperscalers, we would opt for AWS, as compared to other major hyperscalers in the market (Microsoft Azure and GCP), AWS offers the widest range of services and is a established and trusted provider [23], making it the most suitable option for our system. In order to achieve this transition, some changes will be required in the modules and features of our current architecture, as follows:

- **Presentation Layer**:
  - Display Service: AWS Amplify for hosting and AWS Cognito for authentication;
- **Business Layer**:
  - Calendar, Property and User Services: Amazon API Gateway to publish, maintain, monitor, and secure APIs at any scale;
  - Calendar DB and User DB (Relational DB): Amazon RDS;
  - Property DB (NoSQL DB): Amazon DynamoDB;
  - Message Queue: Amazon SQS;
- **Backoffice Layer**:

  - Website Wrappers and Analytics Service: Amazon Lambda, since supports event-based actions, such as those handled by the website wrappers and the analytics service;
  - Website Wrappers: Amazon Elastic Container, as website wrappers have low load on a daily basis, with occasional spikes during scheduled events, such as detecting new reservations or properties. Its automatic scaling based on load ensures that it efficiently uses resources and cost management;
  - Id Mapping DB (Relational DB): Amazon RDS;
  - Analytics DB (Elasticsearch): Amazon Elastic Cloud (EC2);
  - Logging (currently done by Logstash) - Amazon Simple Storage Service (S3);

- **General considerations**:
  - Load Balancing: AWS Auto Scaling;
  - Security: AWS Identity and Access Management (IAM) to control access to resources;
  - Management: Amazon CloudWatch to monitor the performance and health of AWS services;
  - Execution Environment - AWS App Runner, since that everything runs within containers;

This approach will give us the possibility of exponential scaling and adaptation, maintaining the control over the infrastructure and costs.

Regarding European legislation, AWS offers a range of resources and measures to assist in meeting GDPR requirements. These range from compliance with the CISPE-DPC Code of Conduct to the ability to make a data transfer assessment and, where required, demonstrate compliance to data protection authorities [24]. Given these, we shall, therefore, do all it takes to ensure GDPR compliance, thus keeping the personal data of all our users safe.

13 Network Effects

Exploring network effects is a vital aspect in shaping the success of a system like PropertEase. Before diving into network effects exploration strategies to enhance the system's success, it's important to first consider the network it is inserted into due to its characteristics and ecosystem and the inevitable related positive and negative network effects.

13.1 Positive Network Effects

The system benefits from cross-side network effects that are already present in property listing services. On one hand, it becomes easier to find an accommodation on platforms which contain more properties to choose from. On the other hand, it is increasingly easier to rent a property the greater the user base of a website becomes. On the same line of thought, the expansion of both property listings and hosting platforms to list them on, enhances the value of a system like PropertEase (in aiding property management) and its number of users. Furthermore, as the user base grows the value of the network itself also increases as the price recommendation algorithm utilizes more data to more accurately recommend property prices due to the increase in property listings on partnering listing services and, consequently, in the system. This improvement happens both on the internal price variation component and on the property features ML component, since even though the number of properties won't affect the machine learning model directly (they aren't used for training), given the median of the predictions is used, a higher number of properties will produce more accurate results. Thus, all users will benefit of the better price recommendation algorithm.

13.2 Negative Network Effects

Being a platform whose main selling point is the ease of managing properties that are listed in multiple listing platforms, PropertEase's success is highly dependent on those hosting services. Therefore, the size of its network is directly correlated to the number of users the partnering hosting platforms are bringing to PropertEase. Which also means that the success of the system might be in jeopardy if the supported property listing websites don't have a wide user base or/and are not acknowledged in the property listing market. Moreover, if PropertEase becomes too reliant or dependent on certain listing platforms, changes in those platforms policies or technical requirements could affect in a negative way the system's functionality or popularity. For example, if the system supported only two property hosting platforms and one of them went bankrupt then PropertEase would become a glorified new frontend version of the remaining

service with a few more features. Therefore, measures should be taken in order to minimize these issues. Possible solutions for this point will be revisited in the following subsection.

13.3 Network Effects Exploration Strategies to Enhance the System's Success

Following the previously mentioned considerations about network effects and PropertEase's ecosystem, purpose and business model, three different strategies remain in order to maximize success:

- As the system's user base is directly correlated to the user base on external hosting platforms there should be commanded efforts in order to make agreements with property listings' biggest players, not only because they attract more users and their properties, but because they are less likely to experience disruptive difficulties - which meets the following point: the system should support a wide range of hosting platforms in order to mitigate the impact of them facing financial issues, technical issues or reductions on its user base;

- Additionally, efforts should be commanded in order to add value to the system that is independent of these hosting platforms. That can be done through implementing new features that are exclusively dependent on PropertEase. However, the system will benefit more from network effects if PropertEase allows, through its dashboard, the utilization of niche features related to property management that are offered by other external companies' services. These features would include, for example, integrations with cleaning services or automatic locks and surveillance systems for the user's properties.

- As mentioned in Section 3, we have chosen the Freemium model as our business strategy. This model includes a payment barrier in order to gain access to advanced features for a premium price. By implementing a share feature that rewards the user access to those features in a free one-month trial benefits both the company - with a wider user base through sharing - and the users - which can use premium features for free.

14 Conclusion

With tourism beginning to pick up from the pandemic, systems like PropertEase are in higher demand than ever. PropertEase helps property owners manage different listings on multiple platforms by automatically synchronizing bookings and property information, such as price, on the different platforms their properties are listed in. It also offers a price recommendation algorithm that automatically adapts to the market and search trends and has many other property management features.

This paper allowed us to develop a complete business strategy for PropertEase that focused on the critical aspects of placing it in the market. We looked at ecosystem positioning, monetization strategies, the legal limitations and viability, intellectual property, the need for open-source software, and cybersecurity challenges. We also looked at privacy protection, ethical considerations, artificial intelligence, hyperscalers relationships and the network effects.

Overall, PropertEase has a well-defined business model that effectively navigates legal complexities and positions it strategically in the market.

References

[1] Integrations | Icnea. Last accessed on May 25, 2024.

[2] React License. Last accessed on May 25, 2024.

[3] Vite License. Last accessed on May 25, 2024.

[4] TypeScript License. Last accessed on May 25, 2024.

[5] Flatpickr License. Last accessed on May 27, 2024.

[6] Axios License. Last accessed on May 27, 2024.

[7] DaisyUI License. Last accessed on May 25, 2024.

[8] Nginx License. Last accessed on May 25, 2024.

[9] FastAPI License. Last accessed on May 25, 2024.

[10] RabbitMQ License. Last accessed on May 25, 2024.

[11] MongoDB License. Last accessed on May 25, 2024.

[12] PostgresSQL License. Last accessed on May 25, 2024.

[13] SQLite License. Last accessed on May 27, 2024.

[14] Python License. Last accessed on May 25, 2024.

[15] SerpAPI License. Last accessed on May 25, 2024.

[16] ElasticSearch License. Last accessed on May 25, 2024.

[17] Inside AirBnB - Datasets License. Last accessed on May 27, 2024.

[18] ML Model - Kaggle Notebook License. Last accessed on May 27, 2024.

[19] Firebase License. Last accessed on May 27, 2024.

[20] React Auth Kit License. Last accessed on May 27, 2024.

[21] Helping people in Lisbon to predict Airbnb prices. Last acessed on May 27, 2024.

[22] Inside Airbnb: Home. Last accessed on May 27, 2024.

[23] Tejashwini V, "Whats the Difference Between AWS vs. Azure vs. Google Cloud Platform (GCP)?", May 08,2024. Last accessed on May 29, 2024.

[24] AWS, "GDPR compliance when using AWS services". Last accessed on May 29, 2024.

[25] Future of Life Institute - EU AI Act Compliance Checker. Last accessed on May 29, 2024.

[26] AI Regulation Review: The EU AI Act | Zartis. Last accessed on May 29, 2024.

[27] Recital 26 | General Data Protection Regulation (GDPR). Last accessed on May 29, 2024.