

A teaching case for computer information systems courses: Online application for a property rental business

Qingxiong Ma
University of Central Missouri

Silvana Faja
University of Central Missouri

Someswar Kesh
University of Central Missouri

ABSTRACT

It is important to incorporate practical learning experiences into business school courses to equip students with real-world skills for their future careers. This article presents a teaching case designed for Computer Information Systems or similar programs, which can be used for both undergraduate and graduate courses. The case provides students with the opportunity to apply their classroom knowledge to real-world business projects. Through activities that utilize this case, students can learn various skills, including system requirement identification, system architecture design, database system selection, and modern software development using cloud computing technology. Ultimately, the resulting system can benefit property managers, real estate agents, and landlords in the local community by helping them list their rental properties on the market and manage applications.

Keywords: teaching case, system analysis and design, software architecture, database design

Copyright statement: Authors retain the copyright to the manuscripts published in AABRI journals. Please see the AABRI Copyright Policy at <http://www.aabri.com/copyright.html>

INTRODUCTION

Active learning is an instructional technique that aims at actively engaging students with the course material through strategies such as problem solving, case studies, discussions, and group projects. Mitchell et al. (2017) defined active learning as “one time or ongoing student exercises that are introduced in the classroom to encourage student thinking and participation in an effort to engage students in the learning process”. It has been established as an effective learning strategy for both in person and online computing courses (Paetzold and Melby, 2008). Active learning provides an opportunity to explore and apply more advanced topics that could be otherwise difficult to understand by simply listening to lectures and overall it is a more enjoyable experience for students which could lead to better learning results (Shebaro, 2018).

Case studies have been acknowledged as a valuable active learning tool for teaching systems analysis and design, database design and programming. In addition, they can be a great tool for collaborative and group-based learning. Group project-based learning is a widely used pedagogy in information system courses (Mukherjee et al., 2019). Mitchell et al. (2017) conducted a literature review to identify successful active learning exercises for information systems courses. Collaborative student projects, such as business proposals, industry projects or interactive cases are one of these exercises. These exercises are suitable for ongoing or semester-long projects. Research showed that such activities increase student knowledge, encourage problem solving and critical thinking and increase team skills. For example, Mitchell and Vaughan (2022) implemented team-based learning in a database class and compared its results to instructional methods that did not use this approach. Their study concluded that students make better choices when working together to solve problems and their knowledge and understanding of database concepts improved. In addition, the students enjoyed the experience and found it useful for their learning. Team-based learning is also important to develop “soft skills” such as communication skills, negotiation and conflict management skills, time management, and working under pressure.

This article presents an instructional case designed for computer information system courses to provide students with practical experience in solving real-world business problems, apply classroom knowledge, and contribute to the local community. The case is specifically created for undergraduate and graduate courses in the College of Business computer information systems or information systems program. The project aims to equip students with hands-on skills using cutting-edge technology while utilizing real-world examples. The rest of the article is organized as follows: the teaching case is presented first, followed by description of three possible uses of this case in different information system courses.

THE TEACHING CASE

Case synopsis

MAHLAND LLC is a property rental business located in a small college town, with ownership of several apartments and residential houses available for rent. As the business is planning to expand and increase the number of available properties, the property manager needs a better system to handle the renting process.

Challenges

Currently, MAHLAND LLC utilizes a web-based real estate marketplace platform to facilitate the renting process. However, this approach is not only costly but also inflexible, insecure, and inconvenient.

For example, when a customer is interested in a listed property, the customer is offered to fill an online application form, which is different from the company's application template. The online application form does not include some of the necessary items that that company wants. Additionally, the company's application form requires applicants to provide confidential information such as a copy of their driver's license, bank statements, or payroll stubs. When such confidential data is stored in a third-party system, the company has concerns about data security and privacy. The company wants to ensure that the personal information is completely under their control and kept secure.

The message center in the current platform is not very easy to use with different devices. Sometimes, it is hard to browse the messages with a desktop computer. Currently, the user is limited to viewing the messages listed in the default window only, there is no way to move or drag down to browse more messages if you have more messages or inquiries.

One of the difficulties for the property manager in the listing process is the management of the showing appointments. The appointment scheduling, rescheduling, confirming, canceling, sending reminders and communications are quite confusing. The company wants the message system to have a better integration with the showing appointment system, email, and text messages as well.

Another difficult and tedious work for the manager in the renting process is the applicant screening. Currently, the company is using another online service for this aspect of the process. Each time the manager receives an application with an application fee, the manager has to place a screening order. To place an order, the manager must log in to the third party's website and fill a form with the applicant's basic information. The manager has to wait for an email notification when the order is processed. Then the manager re-logs in the system and checks the screening results to make the rental decisions. This login and relogin of the screening system is tedious, and entering the applicant information is redundant.

The company would like to have a system that can provide features similar to the web-based real-estate marketplace platform, but that is easier to use, provides features that are a better fit with company's processes and procedures and alleviates data privacy and security concerns. The company has reached out to your team to help them develop a new system. During the first meeting with the property manager, he provided the following information about the business processes and the functionalities that he would like for the new system.

Meeting with the property manager

During the meeting, you learned that the property manager has three main responsibilities:

- 1- Oversee the property maintenance, which involves repairs of units at the end of the lease and preparing the unit for the new renters, regular maintenance such as pest control and dealing with issues encountered by renters such as plumbing or electrical problems.

2- Managing the renting process, which involves advertising the property, handling requests for information, showing properties, receiving and reviewing applications and managing application related payments.

3- Managing rentals which involves keeping track of payments, sending reminders or notifications, and generating various reports related to properties and renters.

This project only focuses on the renting process. Below is a list of activities that the property manager typically completes during the renting process:

1. Select a listing plan (cost, duration etc.) and publish the property online
2. Add basic property information in the system
3. Upload property photos
4. Set the rent amount
5. Check customer messages or phone calls to schedule showing appointments
6. Conduct onsite tour and answer potential renter questions
7. Distribute application form
8. Collect application form, support documents, and application fee from interested customers
9. Place screening orders from the third party service and make a decision on the applications
10. Discuss, draft, and process the lease agreement
11. Collect payment such as rents, security deposit, pet fees, etc.
12. Plan tenant move-in and distribute keys
13. Move-in inspection and fill move-in check form
14. Move-out inspection and fill move-out check form when lease expired or terminated
15. Calculate the debit receivable and refund security deposit

The new system

One of the main features of the new system is to allow listing of the properties and display the detailed information for each property. The new system should also allow interested visitors to make an appointment for showing and allows them to fill an online application form. The company also expects the system to provide communication convenience so that customers can leave messages or have online conversations with the manager. The last, but not least important feature of the system is to integrate the applicants background check or screening services provided from third parties.

To help understand the system's requirements, the manager provided the following sample documents (see Appendix 1 for samples or prototypes):

1. Property Listing Page (customer's view)
2. Property Details Page (customer's view)
3. Property Creation Page (manager's view)
4. Property Details Entry Page (manager's view)
5. Property Management Page (manager's view)
6. Listing Creation Page (manager's view)
7. Property Application Form (hard copy)

CASE APPLICATIONS AND LEARNING OBJECTIVES

This case is suitable for both undergraduate and graduate level courses such as Software Engineering, Systems Analysis & Design, Database Design, Software Architecture Design, Full-stack Application Development and information systems capstone course. The rest of the article presents three applications of this case, the learning objectives for each application along with a specific set of tasks that the instructor can assign as course activities.

Application One: System Requirements Documentation in Software Engineering or Analysis and Design courses

System requirements identification and analysis is always the starting point and one of the most important phases in software application development. To help students in achieving a correct and complete understanding of the system requirements, it is important to consider some key tactical points. Firstly, as students have mostly been end-users or customers of business applications, they tend to analyze the system and design the solution solely from this perspective. However, it is also essential to consider the viewpoint of the business client, which is often overlooked. Secondly, since students may lack business experience and domain knowledge, it is imperative to encourage them to reach out to real-world businesses and conduct interviews with stakeholders to gain insights into the business process and key operational activities. Thirdly, adherence to engineering rules and conventions is crucial. Often, students' feature lists include mixed syntax and patterns, such as complete sentences, verb phrases, nouns or complex sentences, which can lead to inconsistency, confusion, and ambiguity.

Case studies are an important technique for such courses that enable students to apply their understanding of analysis and design concepts and modeling tools. Due to the scope of this case, it could be assigned as a group project. This teaching case can be used for a group project that can be designed as an outcome-based group project or as a process-based project (Mukherjee et al., 2019). In the outcome-based project, the focus is on the quality of the deliverables, while in the process-based project the focus is on the process used to create the deliverables. The latter approach could include more frequent deliverables but allows for more frequent feedback and enables students to better understand the process of working on each deliverable.

Table 1 (Appendix) provides a list of suggested objectives and corresponding project tasks that can be completed for this teaching case.

Application Two: Data modeling, selection of data persistence approach and database design

Database and Big Data skills continue to remain one of the core skills for information system programs. A recent survey of IT/IS professionals identified database skills as one of the most important skills among incoming IT and business professionals (Cummings and Janicki, 2020).

As noted by Dadashzadeh (2018), the content of the database management systems course in the business curriculum has remained stable covering conceptual data modeling, relational database design and implementation, structured query language (SQL), application development, and database administration. Mason (2018) conducted an analysis of job advertisements and identified the top 20 technical skills needed by a data engineer. This study

showed that relational DBMS skills were not the top skill any longer, instead Big Data, design skills, Hadoop were at the top of the list. An empirical study by Benats et al. (2021) investigated the use of SQL and NoSQL database models in open-source projects and found that the majority of current database-dependent projects (54.72%) rely on a relational database model, while NoSQL-dependent systems represent 45.28% of the projects. The study also confirmed the emergence of hybrid data-intensive systems where (multi-) database models (e.g., relational and NoSQL) are used together. Polyglot persistence (a new term coined for heterogeneous database systems) is envisioned as the database architecture to be prevalent in the future. They explained that two facts have motivated polyglot persistence: (i) the complexity and variety of data to be managed by software systems, and (ii) a single type of database system does not fit all the needs of an increasing number of systems (e.g., learning management systems, online retail systems, or social networks).

Such heterogeneous systems request that developers master several modeling and query languages. To make things more complicated, there are different types of NoSQL systems representing different data models: key-value, document-based, columnar, and graph-based etc., which can present a challenge for students when selecting the appropriate database for a given application.

Wang and Wang (2023) concluded that inclusion of NoSQL skills in a traditional relational database course should be an imperative task for information systems educators. They also noted that there is limited research on how NoSQL databases can be introduced in the traditional database course. This instructional case offers students a chance to develop their skills in choosing the right database technologies for a polyglot persistence application.

Table 2 (Appendix) shows examples of data categories and storage options for each category. In this table, “RDS” refers to relational database systems. In this illustration, DynamoDB is used as an example of a NoSQL database. DynamoDB is a fully managed database service from Amazon, that is offered as part of the Amazon Web Services (AWS) portfolio. In addition, document-oriented databases such as MongoDB and AWS DocumentDB are used to illustrate alternatives to relational database systems.

To implement the database design, data access can be prioritized and sorted. During the implementation process, start with essential data. For instance, basic property data can be addressed first, then property details, photos etc. can be added. When used as a class activity, it is suggested to implement partial models in class. The number of tasks to complete in class depends on the time availability and students’ background. Database design, as any other aspect of system design, is an iterative process, so this case could be used as a running case during several classes. Table 3 (Appendix) presents a set of objectives and tasks that could be included in various course activities that use this teaching case.

Please note that the testing and use of NoSQL databases requires students to have some programming skills. Unlike relational databases where SQL can be used to access and test the database design, there is no such standardized data access layer or tool for the NoSQL database. The code and driver used to access NoSQL databases varies from programming language to language. Table 4 (Appendix) presents examples of data access patterns and NoSQL database design with Amazon DynamoDB.

Finally, in addition to tasks involving data persistence selection and database implementation, course activities could include tasks that involve designing additional queries to support various reports generated by the system. Example of such queries are:

1. Display application status for an applicant.
2. Display all listings based on filtering criteria such as number of bedrooms, pets allowed or not, and rental rate range.

This case can be assigned as a team work where students and faculty can play different roles. Shebaro (2018) proposed these roles that students can take in a team project in a database course: project client, business rules analyzer/writer and database designer. Students can be assigned to different roles for different major features of the system so that they are exposed to the role at least once. Faculty serves as a facilitator by providing clarifications regarding system requirements and supporting self-learning of advanced concepts needed for the case.

Application Three: System Architecture Design

Architectural design plays an important role in system design and implementation. There are multiple architecture models which are suitable for different use cases. The traditional application architecture is monolithic and typically has 3 tiers: user interface, business/data access logic, and data access (or Model-View-Controller pattern (MVC)). Many enterprise applications were created using the simple 3-tier application architecture. For smaller applications, monolithic architecture is often the best solution. But as an application's requirements grow and the number of users increases exponentially, the monolithic architecture can become problematic. Its main drawbacks are that they tend to be big in size, which leads to slower startup time, and the update of any component will require redeployment of the entire application. With the development of Cloud computing, new architectural patterns are emerging. Containerized microservices and serverless are becoming one of the important features for native-cloud applications.

Microservices are an architectural style that structures the application as a collection of services. Each service can be written in a different programming language and tested separately. They are independently deployable and organized around business capabilities. Data are fully decoupled from the application. One of the best choices for creating and running microservices application architectures is by using containers. Microservices architecture involves structuring an application into smaller, independent services that communicate with each other via APIs. This approach allows for scalability and resilience to errors.

In event-driven serverless architecture, the cloud providers such as AWS, Google Cloud, and Azure are responsible for managing the underlying infrastructure, including servers, operating systems, and databases. The developers only focus on writing the application logic, or functions, that respond to specific events or requests. The applications can be defined as a workflow of event-triggered functions, and these functions are executed in a stateless and ephemeral environment that scales automatically based on demand.

However, each design pattern or architecture has its pros and cons. The architectural design adoption decision depends on the application needs. Frequently, the application's non-functional requirements such as security, availability, performance, budget etc. have an important impact on the architectural decision. Sometimes, designers may have to creatively come out with a "hybrid" structure.

This teaching case can be used for architectural design activities in analysis and design courses, software engineering courses, programming courses or a capstone course.

Table 5 (Appendix) presents a set of objectives and tasks related to system architecture design. We recommend developing the application by using the Serverless Monolith approach with AWS Cloud services. This way, students not only learn about the MVC model for small applications, but also become familiar with cloud-based event-driven serverless structure.

Figure 1 (Appendix) shows an example of hybrid architecture using AWS. In this design, MVC is combined with Serverless. The serverless features are characterized by the lambda function and DynamoDB. Each service focuses on one entity such as property management, property photos management, property listing management, property application etc. With this structure, AWS API Gateway acts as the request trigger and forwards all the traffic to the same Lambda function. However, the lambda function in each service has a group of operations. We have to write code to logically process each operation based on the event path and event method (GET, POST or DELETE) passed from API gateway. If we implement each operation in a separate Lambda function, the number of lambda functions will be too much (overkill) for a small application. However, when the number of functionality and requests grow in the future, we can break the Backend API into smaller chunks and move towards Microservices so that each function only handles one specific job.

SUMMARY AND CONCLUSIONS

This article presented a teaching case that could be used by faculty teaching information system courses. In addition to the case, several application scenarios in software engineering, analysis and design and database courses were described.

This teaching case has been used by authors in software engineering and database courses, and the surveys conducted at the end of the semester showed positive results. Students found that by working on the project, they were able to gain a better understanding of the system analysis and design process, as well as the importance of business domain knowledge in system design. Additionally, the project gave students exposure to new technologies like cloud computing, which many found fascinating and pursued further on their own. Some even took online courses and obtained professional certifications. The case presented in this article also provided an opportunity for students to apply their classroom knowledge and skills to solve real-world business problems and make an impact in their local community, which aligns with the objectives of AACSB.

REFERENCES

- Alves, I. and Rocha C. (2021). "Qualifying software engineers undergraduates in DevOps-challenges of introducing technical and non-technical concepts in a project-oriented course." In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pp. 144-153. IEEE.
- Benats, P., Bobert, M., Meurice, L, Nagy, C, Cleve, A. (2021). "An empirical study of (multi-) database models in open-source projects." *Conceptual Modeling: 40th International Conference, ER 2021, Virtual Event, October 18–21, 2021, Proceedings 40*. Springer International Publishing,
- Cummings, J and Janicki, T. (2020). "What Skills do Students Need? A Multi-Year Study of IT/IS Knowledge and Skills in Demand by Employers", *Journal of Information Systems Education*, Vol. 31 (3), pp. 208- 217.
- Dadashzadeh, M. (2018). "A case study to introduce Microsoft Data Mining in the database course." *Journal of Business Cases and Applications*, Vol 22.
- Demchenko, Y., Zhiming Z., Jayachander S., Koulouzis, S., Shi, A., Liao, X., and Gordiyenko, J. (2019). "Teaching DevOps and cloud based software engineering in university curricula", *15th International Conference on eScience (eScience)*, pp. 548-552, IEEE.
- Govekar, M. and Rishi, M. (2007). "Service learning: Bringing real-world education into the B-school classroom." *Journal of Education for Business*, Vol. 83, no.1, pp. 3-10.
- Mason, R. (2018). "Changing Paradigms of Technical Skills for Data Engineers", *Issues in Informing Science and Information Technology*, vol.15.
- Mitchell, A., Petter, S., and Harris, A. L. (2017). "Learning by doing: Twenty successful active learning exercises for information systems courses", *Journal of Information Technology Education: Innovations in Practice*, Vol. 16, pp. 21-26.
- Mitchell, A., and Vaughan, A. G. (2022). "Implementing team-based learning: Findings from a database class", *Journal of Information Technology Education: Innovations in Practice*, Vol. 21, pp.1-23.
- Mukherjee, A. and Bleakney, S. (2019). "Process-Focused Approach to a Systems Analysis & Design Group Project", *Information Systems Education Journal*, Vol. 17(6), pp. 30-40.
- Paetzold, S. and Melby, N. (2008). "Active Learning Strategies for Computer Information Systems Education in Online Courses", *The Journal of Global Business Issues*, pp. 13-17.
- Pérez, A., Moltó, G., Caballer, M., & Calatrava, A. (2018). Serverless computing for container-based architectures. *Future Generation Computer Systems*, 83, 50-59.
- Roy-Hubara, N., Peretz, S. and Sturm, A. (2022). "Selecting databases for Polyglot Persistence applications", *Data & Knowledge Engineering*, Vol. 137.
- Sokolowski, D., Weisenburger, P. and Salvaneschi, G. (2012). "Automating serverless deployments for DevOps organizations", *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 57-69.
- Shebaro, B. (2018). "Using Active Learning Strategies in Teaching Introductory Database Courses", *Consortium for Computing Sciences in Colleges*, pp. 28-36.

Wang, H. and Wang, S. (2023). “Teaching Tip: Teaching NoSQL Databases in a Database Course for Business Students”, *Journal of Information Systems Education*, Vol. 34 (1), pp. 32-40.



APPENDICES

APPENDIX 1. SAMPLE DOCUMENTS PROVIDED BY PROPERTY MANAGER

1. Property Listing Page (customer’s view)

 \$130,000 3 bds 3 ba 1,820 sqft ... 1046 S Maguire Terrace, Warrensburg, MO 64093 Williams Keller LLC	 \$225,000 3 bds 2 ba 2,130 sqft ... 505 E Young st. Warrensburg, MO 64093 Williams Keller LLC	 \$145,000 3 bds 2 ba 2,130 sqft ... 400 Franklin Ave. Warrensburg, MO 64093 Williams Keller LLC
 \$69,000 3 bds 2 ba 2,130 sqft -...	 \$165,000 3 bds 2 ba 2,130 sqft ...	 \$155,000 3 bds 2 ba 2,130 sqft ...

2. Property Details Page (customer’s view)

Property Details



\$145,000 | 3 bd | 2 ba | 2,130 sqft

Overview
 Days listed **8 days** | Views **2,741** | Saves **109**

Solid All Brick Ranch with Nice Layout - Home has been used as Rental all these years but is in great shape(No Sellers Disclosure to be provided)- Tile Entry Area - Nice Living Room - Huge Eat-in Kitchen with lots of Space for Center Island if wanted to add Plus the Dining Area (Pantry with roll out drawers)-All 3 Bdrms are Good Size - Master Bedroom has nice walk in closet - Patio off Back Sliding Door out the Kitchen - Back Yard is Private with Mature Trees at the Back Line of Property - Does have newer Roof put on within last few years

Price and tax history	Down payment assistance	Similar homes
Monthly cost	Rental value	Neighborhood
	Nearby schools	Local Legal Protections

3. Property Creation Page (manager’s view)

Add New Property

Street Address:

City:

State:

Zip:

Year Built:

Type:

Size in Square Footage:

Number of Bedrooms:

Number of Bathrooms:

Neighborhood:

4. Property Details Entry Page (additional information that customers are interested to know)

Add detailed information to your property

Description:

Example: Owner pays for lawn care and water. Tenant is responsible for gas and electric.




Laundry facilities:

- No laundry facilities are available
- Washer-dryer included
- Washer-dryer hookups
- Shared or in building

Amenities or Features

<input type="checkbox"/> Air conditioning	<input type="checkbox"/> Balcony or deck	<input type="checkbox"/> Furnished
<input type="checkbox"/> Balcony or deck	<input type="checkbox"/> Furnished	<input type="checkbox"/> Pet friendly
<input type="checkbox"/> Furnished	<input type="checkbox"/> Pet friendly	<input type="checkbox"/> Hardwood floors
<input type="checkbox"/> Pet friendly	<input type="checkbox"/> Hardwood floors	
<input type="checkbox"/> Hardwood floors	<input type="checkbox"/> Balcony or deck	

5. Property Management Page (manager’s view)

	Address	Listing Status	Actions
	15513 Glenwood Ave Overland Park, KS 66223	Active Aug 9, 2023 – Aug 21, 2023	View Update Delete
	7012cW 157th Ter Overland Park, KS 66223	Inactive	View Update Delete
	1201 S Maguire St Warrensburg, MO 64093	Inactive	View Update Delete

6. Property Lease Information Page (manager’s view)

Lease Information

Monthly rent:

Security Deposit:

Available Date:

Lease Terms:

Example: Owner pays for lawn care and water. Tenant is responsible for gas and electric.

Pet Policy

No pets allowed

cats allowed

Small dogs allowed

Large dogs allowed

for rent by (your role of listing this property for rent):

Property Owner

Property Manager

Agent or Broker

Tenant

7. Property Application Form

Rental Application Guidelines

APPLICANT –All adult applicants (18 years or older) must complete a separate application for rental. Please provide the following information in full for your application to be processed in a timely manner:

1. **Application fee** - The application processing fee is **\$45 for each applicant. This fee must be paid in cash or certified funds.** This fee is non-refundable. Our Venmo username is: xxxxxxxx
2. **Photo ID** - Provide a copy of a valid state or government issued picture identification for each applicant 18 years and older.
3. **Income Information** - Include name of supervisor, length of employment, telephone number and a copy of one month's most recent pay stub.
4. **Other** - You may also include any other documentation that may support your application, such as benefit statements, letter(s) of employment, letter(s) of recommendation, and bank statements.

Contact information:

Property Manager: xxxxxxxxxxxx

Phone: xxxxxxxxxxxx

email: xxxxxxxxxxxxxx

Note: You will be denied rental if you misrepresent any information on this application. If misrepresentations are found after a rental agreement is signed, your rental agreement may be terminated at Landlord's discretion.

Thank you for considering renting from us.



RENTAL APPLICATION

Today's Date _____ Address Applying For _____ Move-In Date _____

Applicant Name: _____ SS#: _____ Date of Birth: _____ Email: _____

List all other persons to occupy apartment that are under 18 years old

Name: _____ SS#: _____ Date of Birth: _____ Email: _____

Name: _____ SS#: _____ Date of Birth: _____ Email: _____

Current Employment of Applicant

Employer _____

Address _____

Phone _____ Length of Time _____

Position _____ Supervisor _____

Approx. Income \$ _____ wk. mo. yr

Former Employment of Applicant

Employer _____

Address _____

Phone _____ Length of Time _____

Position _____ Supervisor _____

Approx. Income \$ _____ wk. mo. yr

Other Income : _____ **Source:** _____

Pets: YES NO If yes how many? _____ What Kind? _____

Present Street Address: _____

City / State / Zip: _____

Length of Time: _____ Owns Rents Do you have a lease? _____ Expires When? _____

Name of Landlord or Mortgage Holder: _____ Phone No: _____

Previous Street Address: _____

City / State / Zip: _____

Length of Time: _____ Owns Rents Do you have a lease? _____ Expires When? _____

Name of Landlord or Mortgage Holder: _____ Phone No: _____

Have you ever been evicted or foreclosed from any premises? Yes No

If yes, explain: _____

Nearest Relative (Other than Husband or Wife) – WHO TO REACH IN AN EMERGENCY:

Name: _____ Relationship _____ Phone: _____ Email: _____

Address: _____ City/State/Zip: _____

FALSE INFORMATION GIVEN ON AN APPLICATION IS IN ITSELF GROUNDS FOR REJECTION OF THE APPLICATION OR TERMINATION OF TENANCY.

Authorization for Release of Information

I authorize without reservation, any party (including, but not limited to, employers, law enforcement agencies, state agencies, institutions and private information bureaus or repositories) contacted by prospective property manager or property owner to furnish any or all of the above-mentioned information. I release and discharge all liability from all companies, agencies, officials, officers, employees and other persons, who, in good faith provide to prospective property manager or property owner the above-mentioned information as requested, in order to successfully complete a background investigation of my rental application. I will allow a photocopy of this authorization to be as valid as the original.

Date: _____ Applicant's Signature: _____

Home Phone: _____ Work Phone: _____

Equal Housing Opportunity

APPENDIX 2. FIGURES AND TABLES

Objectives	Tasks
1. Understand the requirements elicitation process	1.1 Identify key stakeholders in this case. 1.2 Identify requirements gathering techniques that could be used in this case and describe the process and stakeholders involved.
2. Become familiar with various techniques to document system features.	2.1 Create a set of user stories to describe system functionalities. 2.2 Create use case narratives. 2.3 Create a formal system requirements specifications document.
3. Use modeling techniques to document system features	3.1 Create a functional model of system features using a use case diagram
4. Use prototyping as a tool to verify system requirements and apply user experience design principles and best practices.	4.1 Create prototypes of user interfaces, screens and reports, to reflect functionalities described in the other deliverables. 4.2 Describe any additional requirements identified during prototyping or any modification to those previously identified. 4.3 Make sure to apply user interface design principles discussed in the course such as consistency, ease of use, ease of learning.
5. Analyze the problem domain and identify the entity classes.	5.1 Identify entity classes and create a class diagram to model these classes and their relationships.
6. Apply principles of agile development or any other process chosen by the instructor	6.1 Develop a project plan that includes a series of iterations. 6.2 Identify tasks to be completed during each iteration, time estimations and roles that team members would play during this process.
7. Understand the relationship between the implementation phase and other phases of the system development	7.1 Explore implementation strategies by evaluating different tools and platforms that could be used to develop the system.

Table 1. A set of objectives and tasks related to analysis and design activities

Data category	Types and access requirements	Storage technologies options
Basic property data	Factual numbers and short text Accessed frequently and requires higher performance	RDS or DynamoDB
Property details	Long text-based description and unstructured data about property amenities or features	Document-based NoSQL such as MongoDB or AWS DocumentDB
Property photos/videos	Multimedia, big, binary files	File systems such as S3
Property leasing data	A mix of numbers and text such as rent, security deposit, pet policy, lease terms Accessed frequently and requires higher performance	RDS or DynamoDB
User profile	Short text such as name, address, contact formation, and login credentials	RDS or AWS Cognito User Pool
Property showing appointments	Structured data such as date, time, potential renter name	RDS or DynamoDB
Rental applications	Unstructured, mixed data	Document-based NoSQL such as MongoDB or AWS DocumentDB
Messages	Short text with categories (listing, viewing, application) and timestamps Requires high performance	RDS or DynamoDB
Contracts	Unstructured, mixed data	Document-based NoSQL such as MongoDB or AWS DocumentDB
Attachments	Multimedia files	File systems such as S3

Table 2. Illustration of data storage options

Objectives	Tasks
1. Analyze data	1.1 Analyze deliverables from Application 1 such as prototypes, functional requirements, use case diagrams, and class diagrams. 1.2 Identify and list all entities and their attributes. 1.3 Create an entity relationship diagram.
2. Determine data storage systems	1.1 Analyze data types and application access requirements 1.2 List data storage options 1.3 Compare options (performance, cost, security, availability etc.) 1.4 Choose the appropriate storage options
3. Create conceptual data model (for RDS only)	3.1 Refine the entity relationship diagram in step 1. 3.1.1 Identify and list all functional dependencies. 3.1.2 Normalize the data and bring all the tables you come up with into at least 3NF, and to BCNF and/or 4NF where applicable. 3.2 Identify and include cardinalities (both maximum and minimum) based on the information provided. 3.3 Create a data dictionary based on your entity relationship diagram and normalization solution.
4. Create logical database design (database schema)	RDS only: Transform the conceptual model into database design: 4.1 Identify and list all primary keys and foreign keys for all tables you have identified in your normalization solution. 4.2 Specify the SQL server data types and other attributes for each of the columns in every table you have identified. 4.3 Document the minimum cardinality enforcement approach for relationships between each pair of the tables. NoSQL: 4.1 Design the key and conditions based on application data access patterns
5. Database implementation	In the selected RDS: 5.1 Create the tables as specified in the database schema 5.2 Create the SQL statements for each query 5.3 Map each data access pattern to a specific query or program code. In the selected NoSQL system: 5.1 Create the tables or collections 5.1 Provide methods or functions used for testing
6. Test the data access functions	6.1 Connect to the database system programmatically to test each data access function

Table 3. Objectives and tasks related to database course activities

Access Patterns	Table/GSI	Key Conditions	Method
Add a property	PropertyBasic	PK=pid and SK = username	put_item()
Get a property	PropertyBasic	PK=pid and SK = username	get_item()
List all properties	PropertyBasic		scan()
Delete a property	PropertyBasic	PK=pid and SK = username	delete_item()
Update a property	PropertyBasic	PK=pid and SK = username	update_item()

Table 4. Examples of access patterns in a NoSQL database

Objective	Tasks
1. Understand system requirements	1.1 Analyze deliverables from Application 1 such as prototypes, functional requirements, use case diagrams, and class diagrams. 1.2 Identify and list all read/write operations (services) 1.3 Review non-functional requirements such as availability, performance, security, cost etc.
2. Determine the architectural design pattern	2.1 Compare different architectural design options such as MVC, microservices, event-driven serverless 2.2 Determine an architecture that can better meet the system requirements both functional and non-functionally 2.2 Create an architectural diagram to show the components and relationships
3. Modify and optimize the design pattern	3.1 incorporate new identified features, or refactoring and restructuring the design pattern 3.2 Re-draw the architectural diagram
4. Test the system	4.1 Design test cases and metrics for both functional and non-functional requirements 4.2 Run the system and check the performance metrics such as the throughput, throttles, security breaches etc.

Table 5. Objectives and tasks related to system architecture design activities

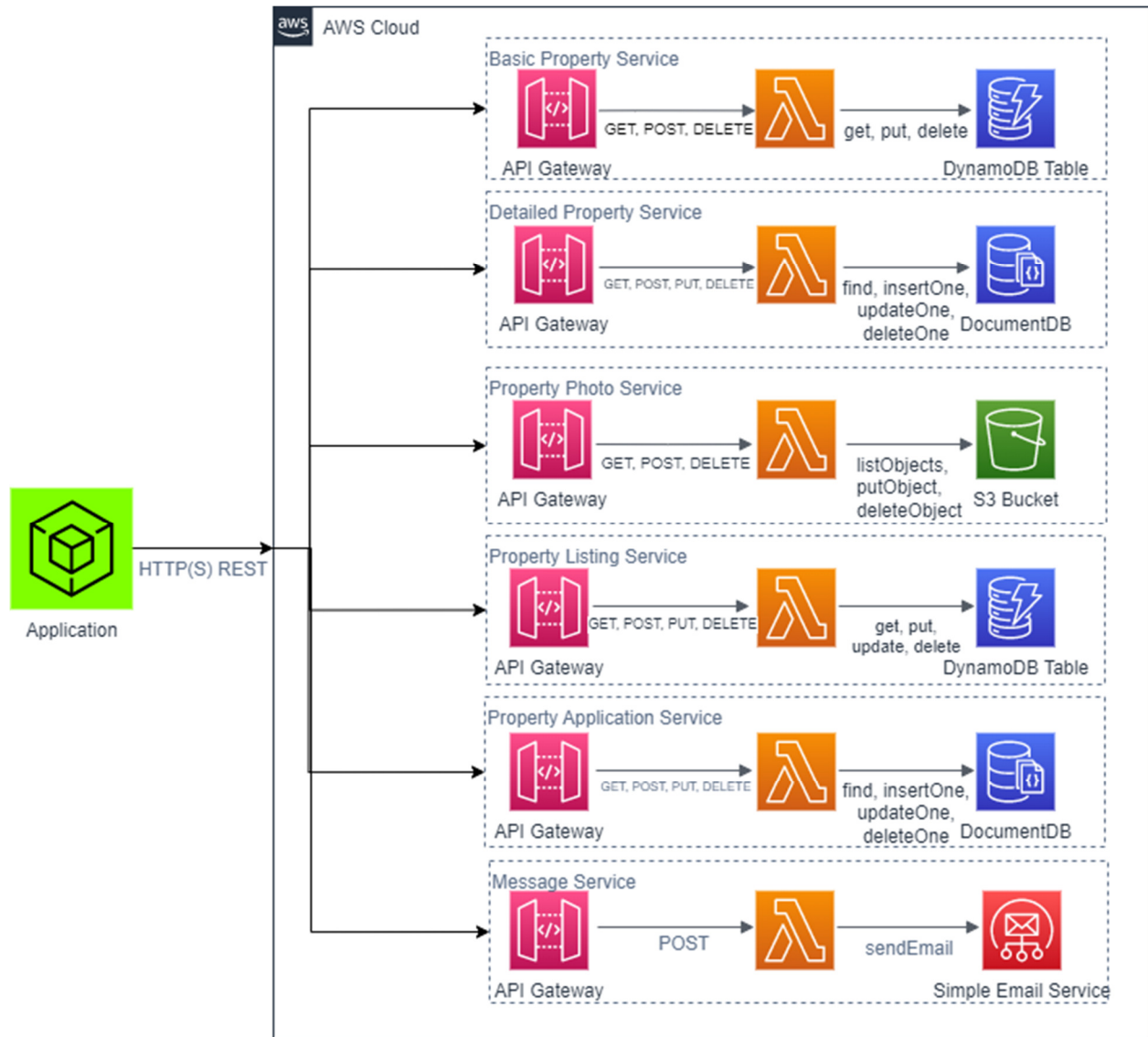


Figure 1. Example of hybrid architecture using AWS