Contents

[**1.** **Introduction.** 2](#_Toc52229668)

[**2.** **The 3-Tier Architecture.** 2](#_Toc52229669)

[**3.** **Design Architecture** 5](#_Toc52229670)

[**4.** **Use Case Diagram** 8](#_Toc52229671)

[**5.** **Class diagram.** 8](#_Toc52229672)

[**6.** **Typical Workflow.** 10](#_Toc52229673)

[**7.** **End-User Manual.** 11](#_Toc52229674)

[**7.1.** **Download and install JDK.** 11](#_Toc52229675)

[**7.2.** **Download and Install Netbeans IDE 8.2 RC.** 11](#_Toc52229676)

[**7.2.** **Install Apache Derby.** 12](#_Toc52229677)

[**7.3.** **Install Glassfish Server.** 13](#_Toc52229678)

[**7.4.** **Compiling project.** 15](#_Toc52229679)

[**7.5.** **Deploy project.** 17](#_Toc52229680)

[**8.** **Test Instruction.** 19](#_Toc52229681)

# **Introduction.**

Real estate company is currently running a business on properties and the CEO has decided to request an IT service company to provide e-business system to improve business efficiency. Capturing requirements and budget analysis were performed and one important requirement is to implement 3-tier enterprise application in the system. The decision has been made is to apply Java EE open source technology and web base to develop the system.

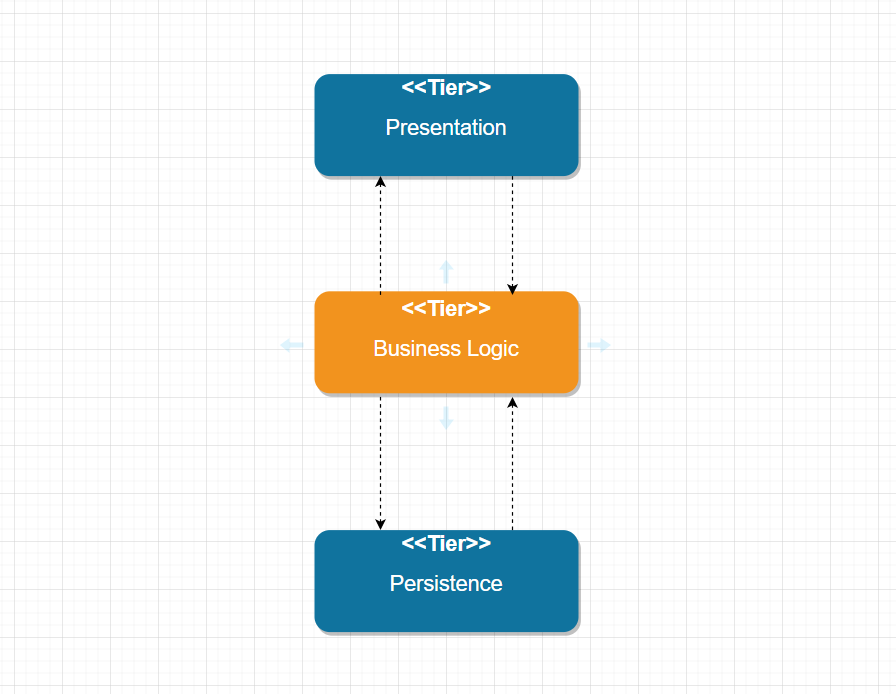
The purpose of this report is to provide information to users about what the 3-tier architecture is, how this architecture is applied in the system, and how the system works.

The system is to provide flexibility to manage properties across all platforms because of Java EE technologies. There are three main features: property management, property manager and allocations management.

This report is divided into four sessions: first is to generally discuss 3-tier architecture, second is to show the typical workflow of the system, third is to provide end-user manual and final is test plan.

# **The 3-Tier Architecture.**

The 3-tier architecture is software engineering development approach that is divided into three-tier separately and independently. This architecture gives the ability to indecently modified and replace each tier without impact other tiers. Therefore, the development team can focus on one tier only if there are changes in one tier only compare with other architecture. The 3-tier architecture is integrated by a presentation tier, a business tier, and a persistence tier as shown below:



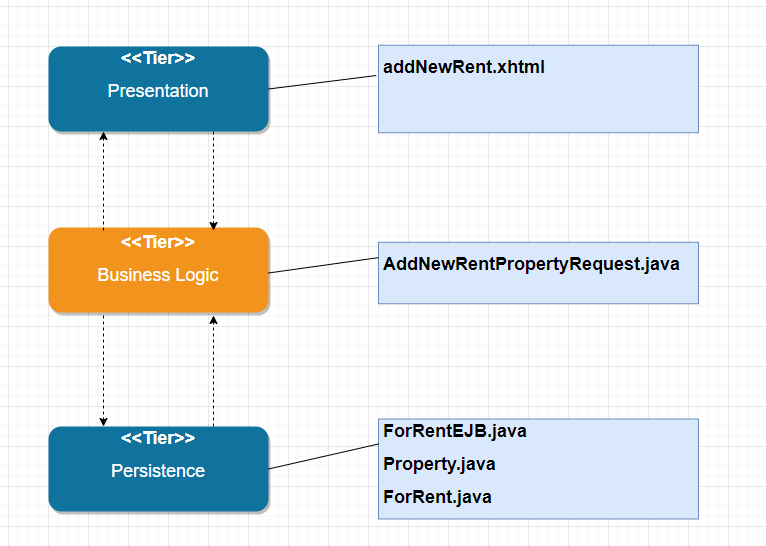
**Picture 1: 3-tier architecture.**

* The presentation tier is to present data to user mostly graphical way and it is a boundary between system and user. This tier also helps the user to communicate with the system and it also communicates the business logic tier. Presentation tier only talks to business logic tier and user.
* Business logic tier is holding all business logic for an application and it is code that provides functionality to meet requirements. This tier only communicates to the presentation tier and the persistence tier.
* Persistence tier is containing all database structures, connection to have the ability to insert, update, delete by using queries, expressions and transactions. This tier only communicates database managed system and business logic tier.

*Base on the concept of 3-tier architecture, the application is designed as shown below:*

|  |  |
| --- | --- |
| Tier | Artefact |
| Presentation | addNewAllocation.xhtml  listAllAllocation.xhtml  searchAllocation.xhtml  footerPage.xhtml  headerPage.xhtml  layoutPage.xhtml  menuPage.xhtml  addNewManager.xhtml  listAllManagers.xhtml  searchManager.xhtml  viewManagerDetails.xhtml  addNewRent.xhtml  listAllRent.xhtml  searchRent.xhtml  viewRentPropertyDetails.xhtml  addNewSale.xhtml  listAllSale.xhtml  searchSale.xhtml  viewSalePropertyDetails.xhtml  compositeProperty.xhtml  compositeViewProperty.xhtml  default.css  errorPage.xhtml  index.xhtml |
| Business logic | ErrorController.java  FunctionalClass.java  IndexController.java  SendMSG.java  SessionController.java  AddNewAllocationRequest.java  ListAllAllocationRequest.java  SearchAllocationRequest.java  AddNewManagerRequest.java  ListAllManagersRequest.java  SearchManagerRequest.java  ViewManagerDetailsRequest.java  AddNewRentPropertyRequest.java  ListAllRentProperyRequest.java  SearchRentPropertyRequest.java  AddNewSalePropertyRequest.java  ListAllSalePropertyRequest.java  SearchSalePropertyRequest.java |
| Persistence | ABaseBean.java  AllocationEJB.java  ForRentEJB.java  ForSaleEJB.java  PropertyManagerEJB.java  Allocation.java  AllocationID.java  ForRent.java  ForSale.java  IBaseEntity.java  Property.java  PropertyManager.java |

*Example of saving new rent property:*



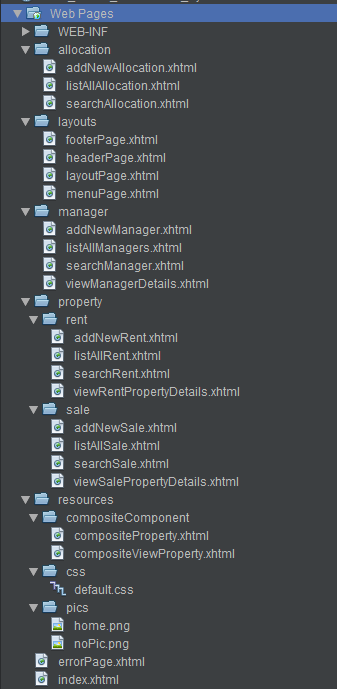
**Picture 2: Case of saving new rent property**

The process of saving new rent property:

* Presentation tier capture input from user send it to business logic tier
* Business logic tier receive data from presentation tier, process it and send it to persistence tier
* Persistence tier received data from the business logic tier and open connection to the database managed system to save the new rental property to the database

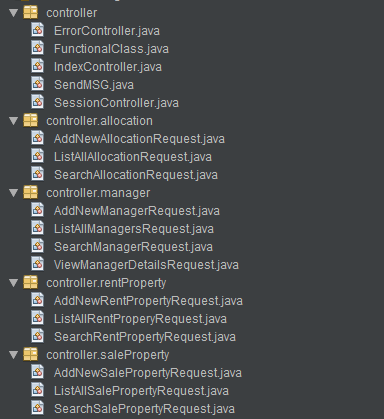
# **Design Architecture**

*Presentation tier:*



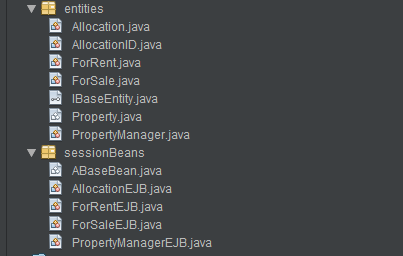
**Picture 4: Presentation tier**

*Business logic tier:*



**Picture 5: Business logic tier**

*Persistence tier:*



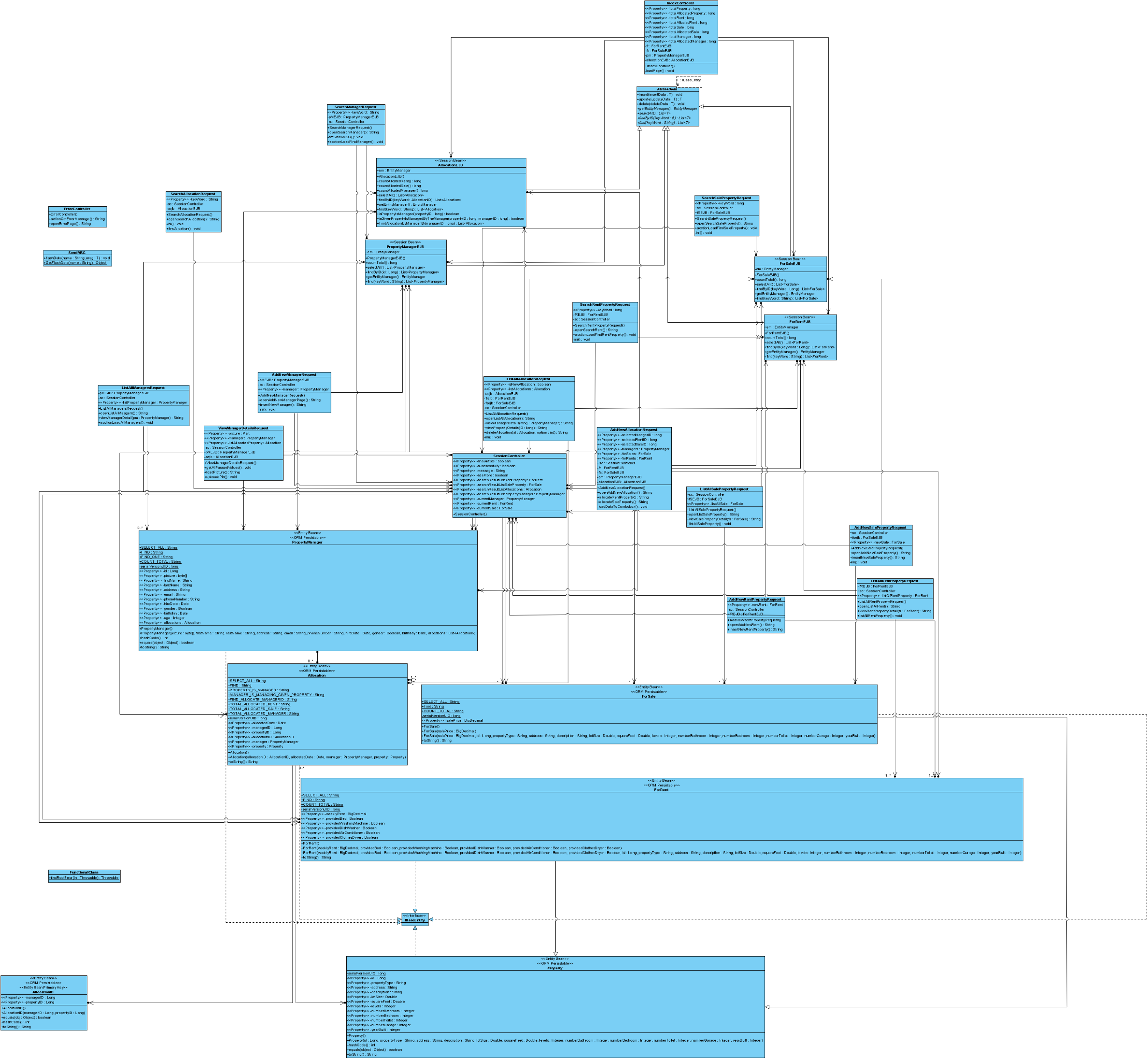
**Picture 6: Persistence tier**

# **Use Case Diagram**



**Picture 7: Use case diagram**

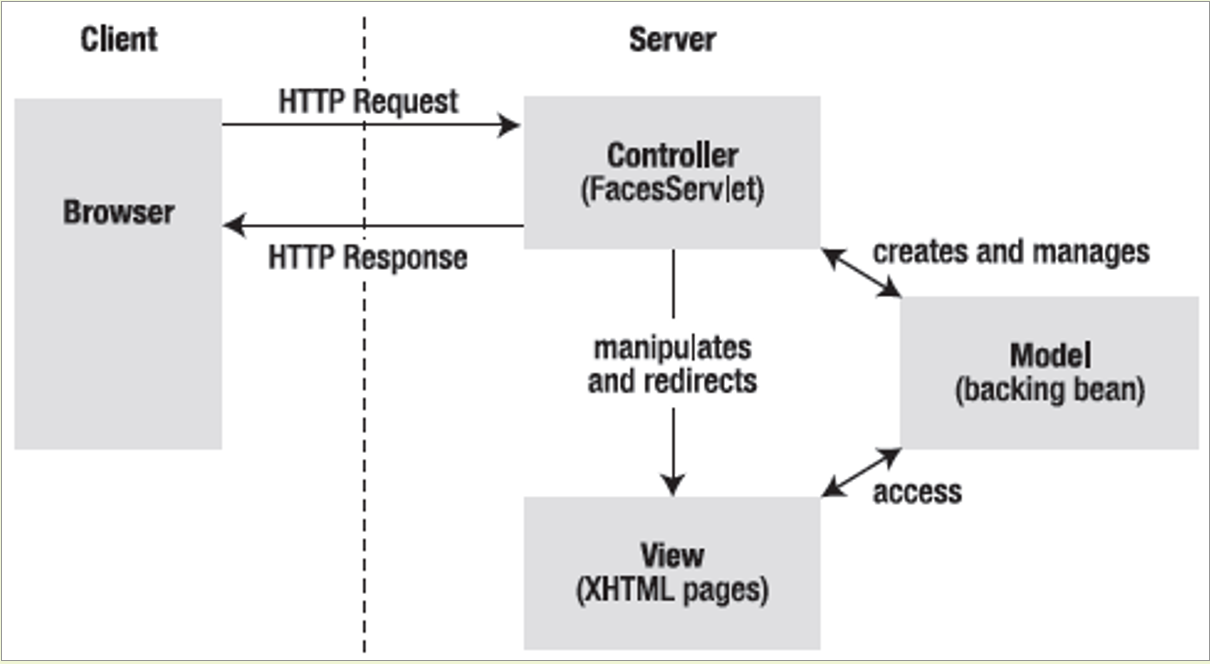
# **Class diagram.**



**Picture 8: Class diagram of the whole system (See Class diagram.png file if it is too small)**

# **Typical Workflow.**

In a lecture, COIT20259 Enterprise computing architecture, presented at CQUniversity, Brisbane Campus, on 25 September 2020, Mahesh suggested diagram as workflow below:



**Picture 9: Workflow architecture.**

In general

* In client-side, the user using the web browser to communicate with server-side by sending an HTTP request and the server response back the request as an HTTP response.
* In the server side, when the request is received, the controller process the request and talk to model or redirect to another view. The model talks to the database managed system to retrieve data or it can access to the view to capture input from the user.

Base on the diagram, the workflow of add new rental property will be listed below:

* In client-side:
  + The addNewRent.xhtml is shown, the user entering property details then data will be sent to server-side under HTTP request
* In the server-side, the controller (AddNewRentPropertyRequest.java) receives the request, then process and talk to model (ForRentEJB.java, ForRent.java and Property.java) to open a connection to the database managed system to insert new rental property. Next, the controller will redirect another page (viewRentPropertyDetails.xhtml) to the view details of the rental property by sending the HTTP response to the client side.

# **End-User Manual.**

## **Download and install JDK.**

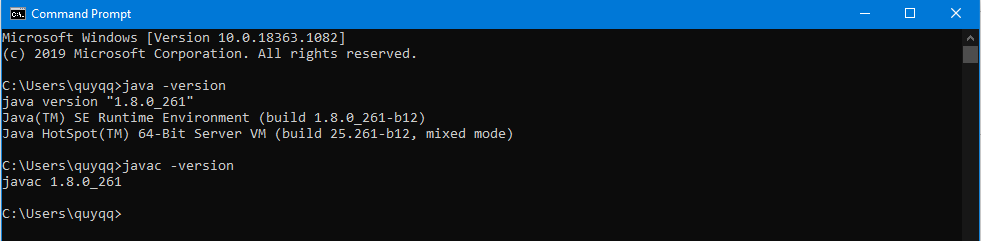
Download JDK by access this link: <https://www.oracle.com/au/java/technologies/javase/javase-jdk8-downloads.html> (this application was developed on Window platform)

Follow the instruction of installation then check the JDK is installed successfully follow steps below:

* Open command prompt
* Typing: Java –version
* Hit enter
* Typeing: Javac –version
* Hit enter

**Setting up system by accessing the link** <https://docs.oracle.com/cd/E19182-01/821-0917/inst_jdk_javahome_t/index.html>

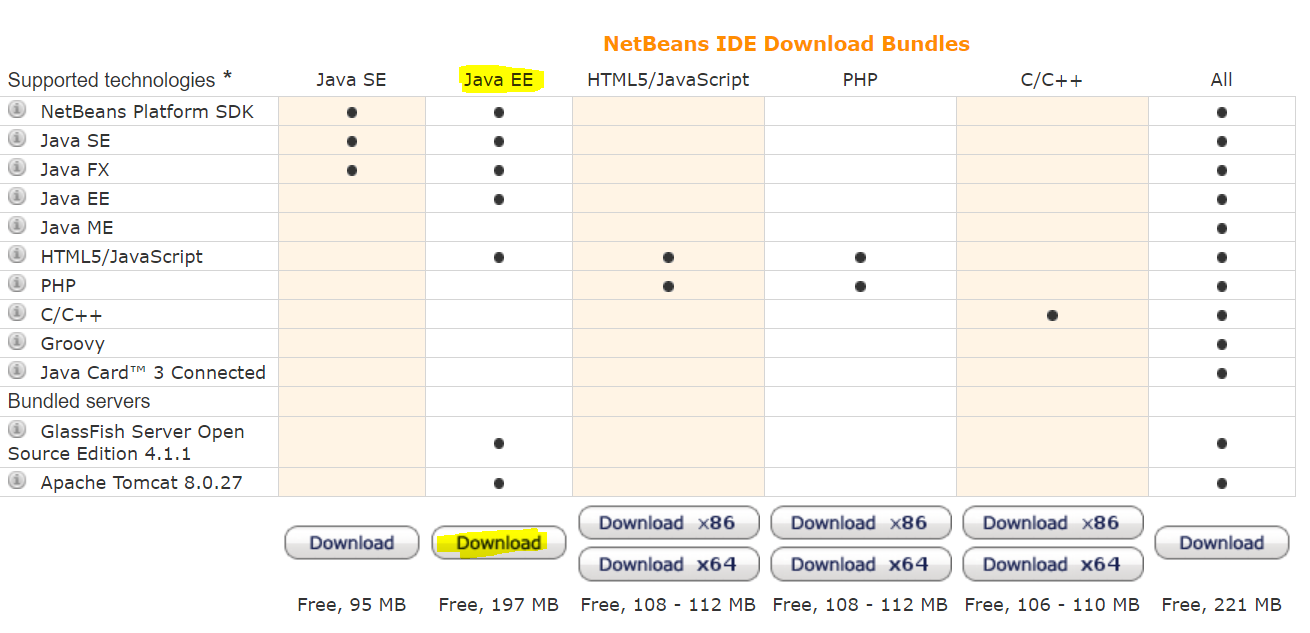
The result is shown below if the installation is correct. Java and Javac version have to be the same. This project is using JDK 8.



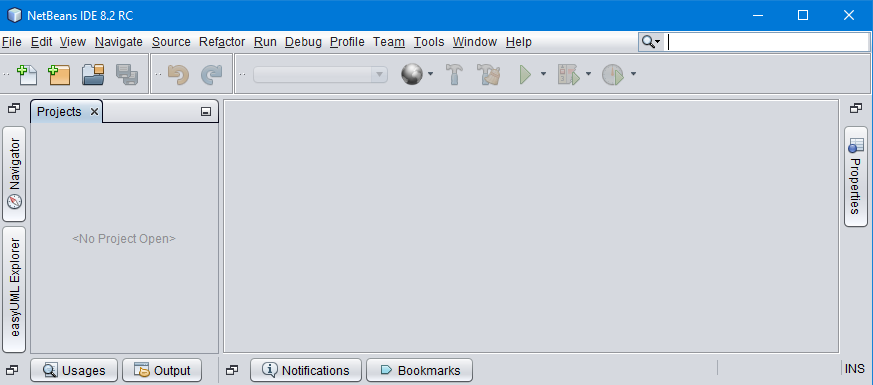
## **Download and Install Netbeans IDE 8.2 RC.**

Download **Netbeans IDE 8.2 RC** by access this link: <https://netbeans.org/downloads/8.2/rc/>

Download the version as shown below:

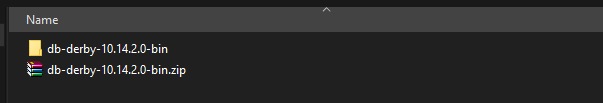


Follow instruction of installation then fire up **Netbeans IDE 8.2 RC** with successful screen shot below:



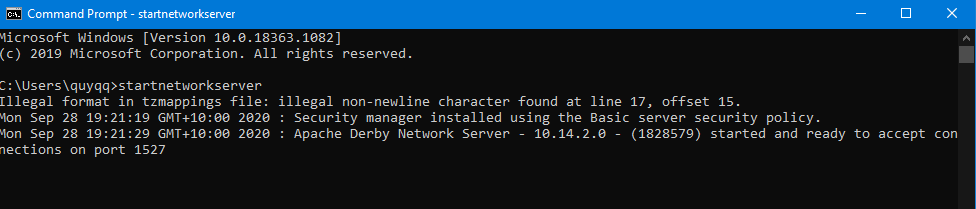
## **Install Apache Derby.**

Download Apache Derby by access this link: <https://db.apache.org/derby/derby_downloads.html> (download version 10.14.2.0 ) and unzip the file in a folder as shown below:



Follow installation instruction of this link: <https://db.apache.org/derby/papers/DerbyTut/install_software.html>

If anything setup correct, open command prompt and type **startnetworkserver** the result will be shown below:

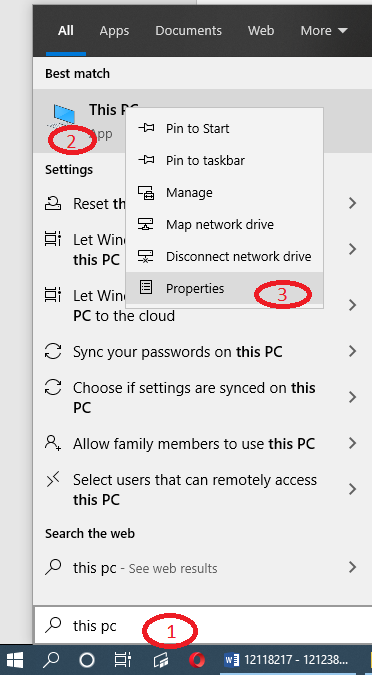


## **Install Glassfish Server.**

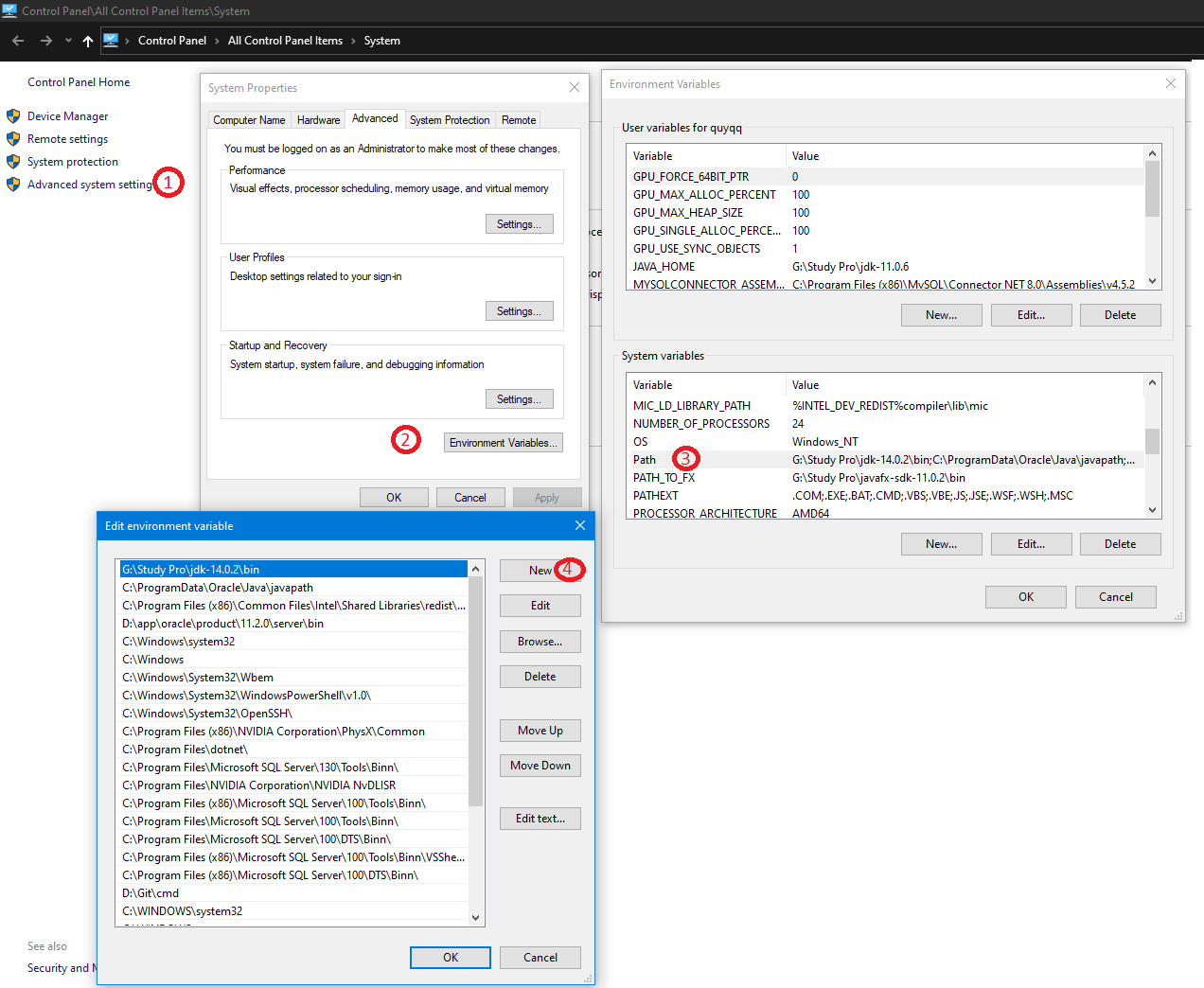
Download Glassfish sever by access this link: <https://javaee.github.io/glassfish/download> (recommend to download version 4.1.2 full platform ) unzip the file as shown below:



Click on window icon bottom left, type “this pc” and right click on this pc then select properties as shown below:



Keep going with the picture below:

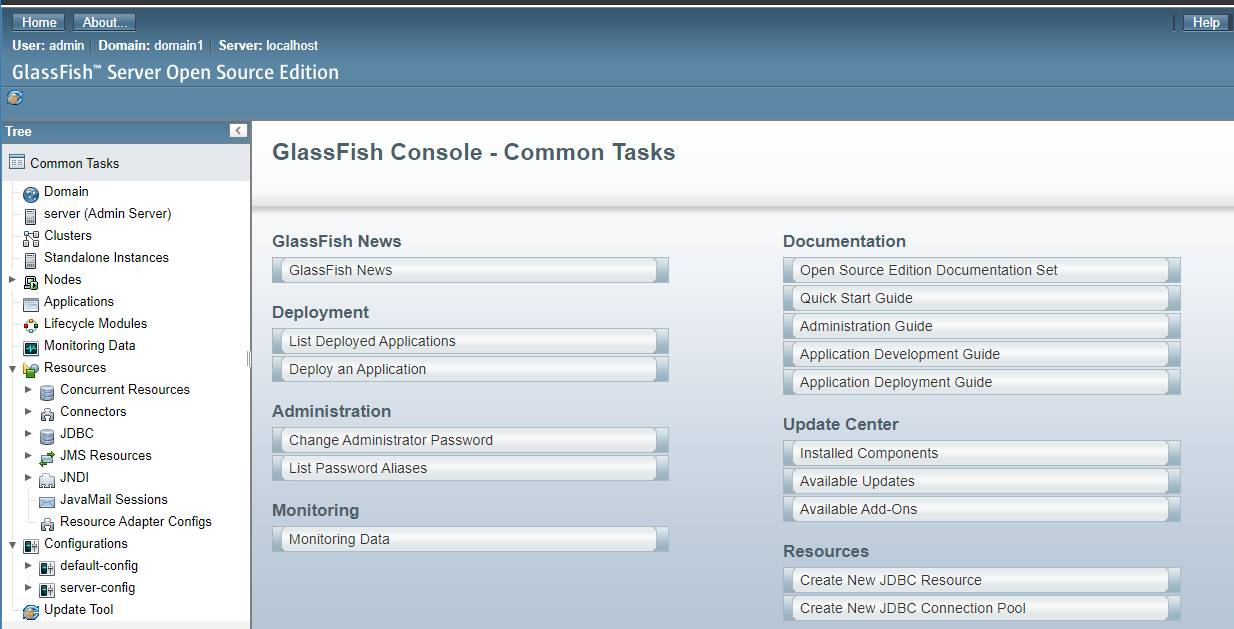


Then add two line with your path to Glassfish4 folder:

* <your path>/glassfish4\bin
* <your path>/glassfish4\glassfish\bin

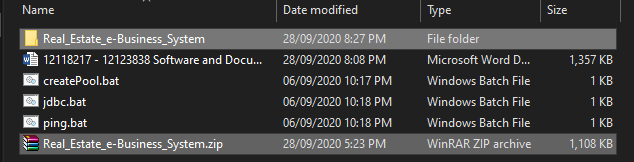
Check server is running correctly by typing: **asadmin startdomain** in command prompt and access <http://localhost:4848/>

The result as shown below:

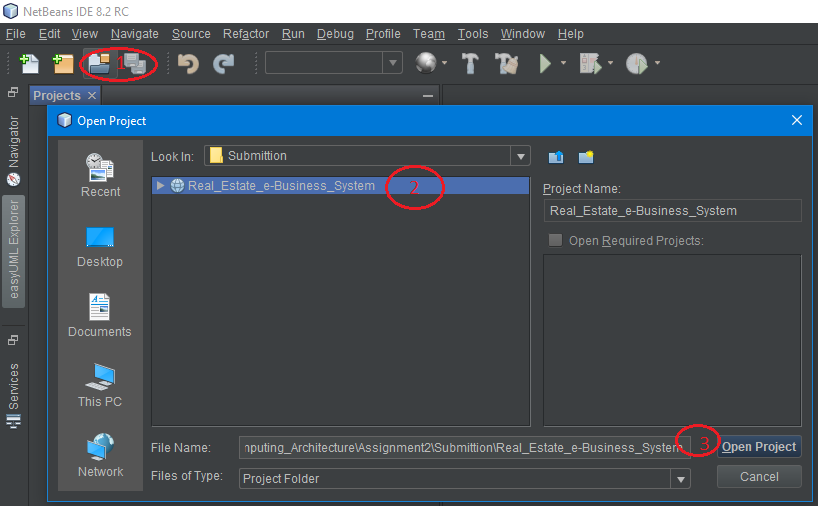


## **Compiling project.**

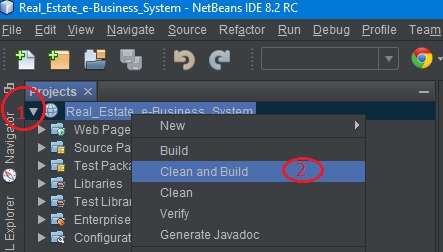
Unzip file **Real\_Estate\_e-Business\_System.zip** as show below:



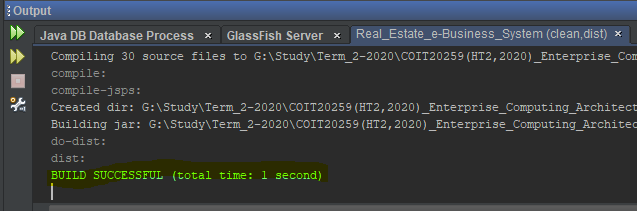
Open **Netbeans IDE 8.2 RC** and flow picture as show below:



Right click on project and click clean and build as shown below:



Result:

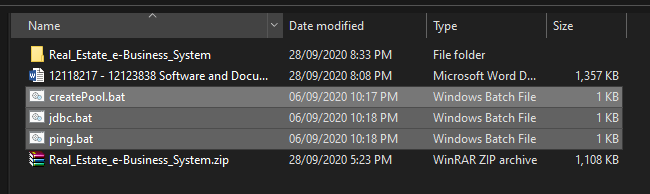


## **Deploy project.**

Make sure Apache derby and Glassfish are running first.

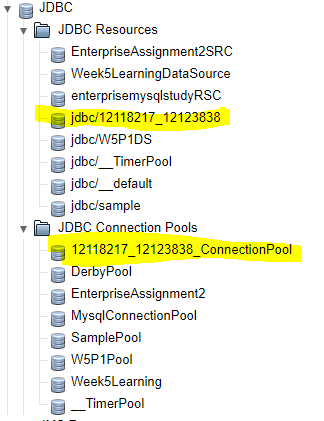
Run 3 files:

* createPool.bat
* jdbc.bat
* ping.bat

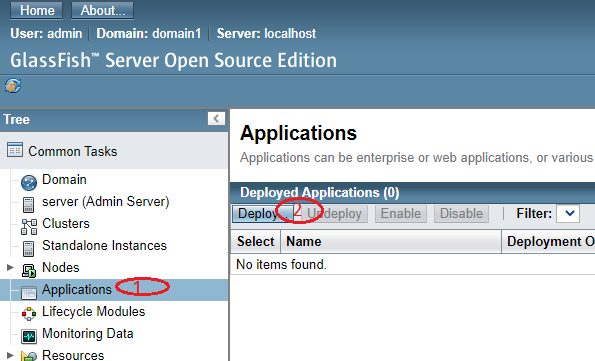


Access to server by access link: <http://localhost:4848/>

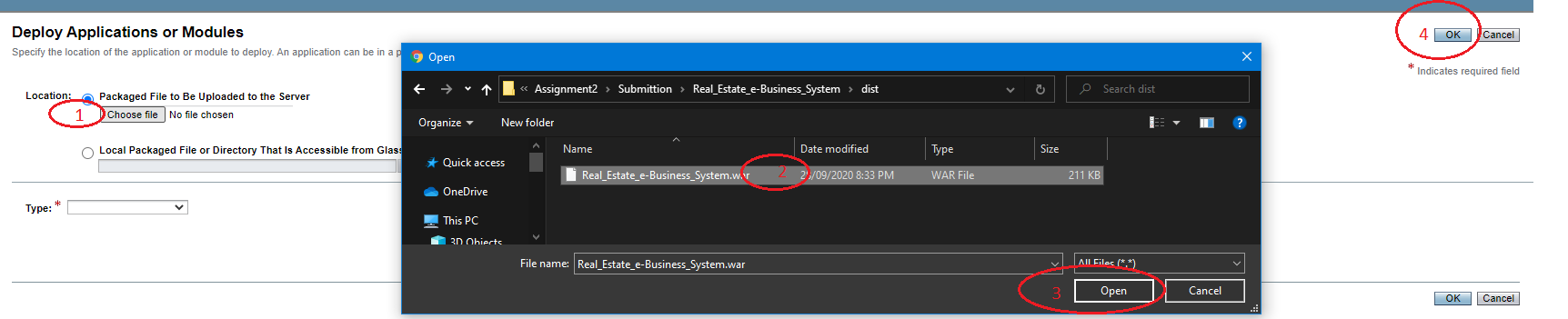
Check connection pool and jdbc are created as show below:



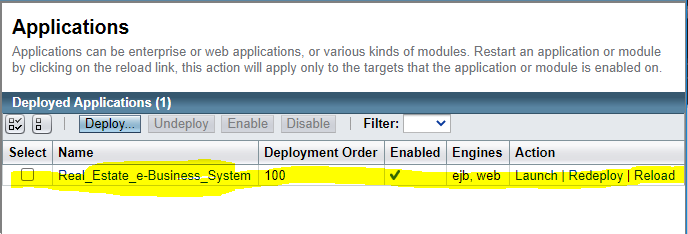
**Deploy project by flowing the picture below:**



Navigate to the project folder **…\ Submittion\Real\_Estate\_e-Business\_System\dist** to select the **file Real\_Estate\_e-Business\_System.war**

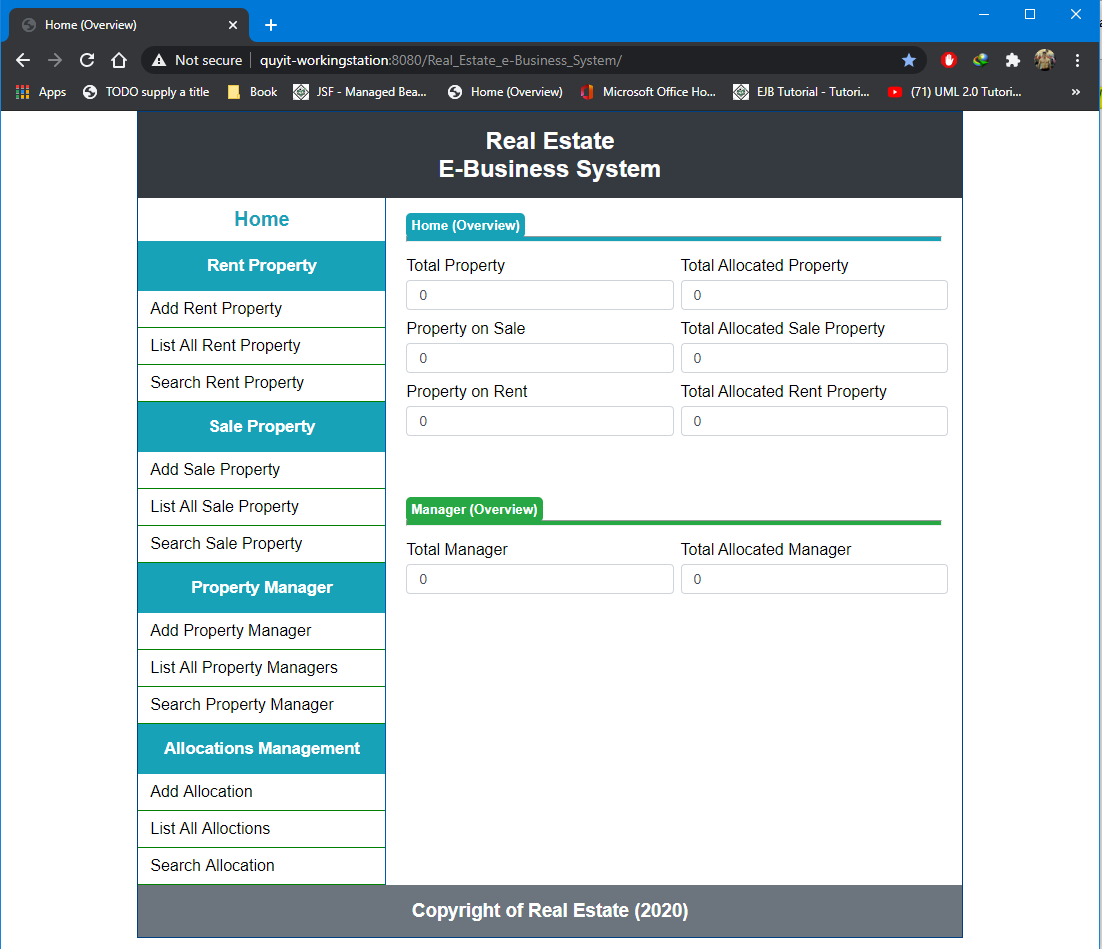


The result successfully deploys:



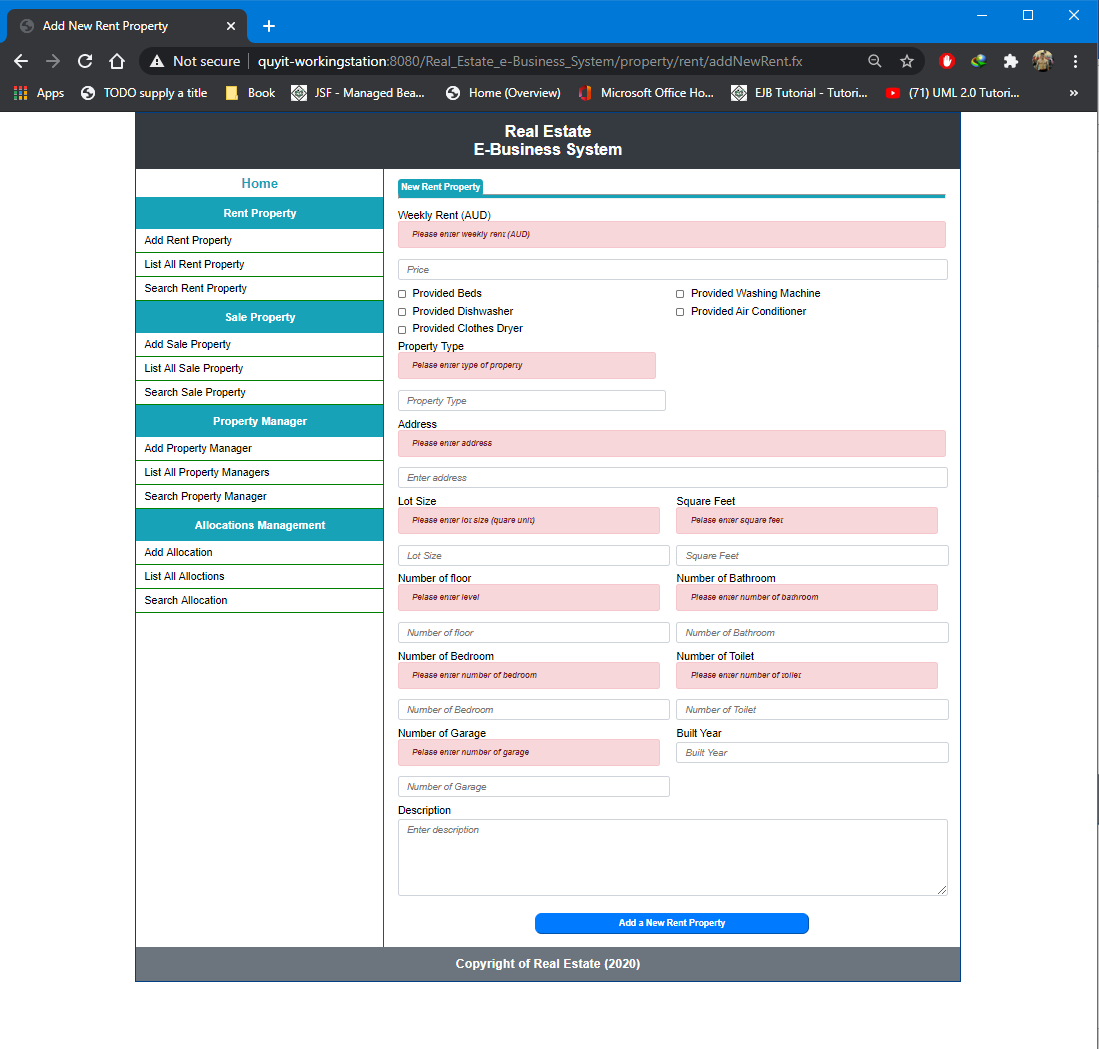
# **Test Instruction.**

**Home page:**

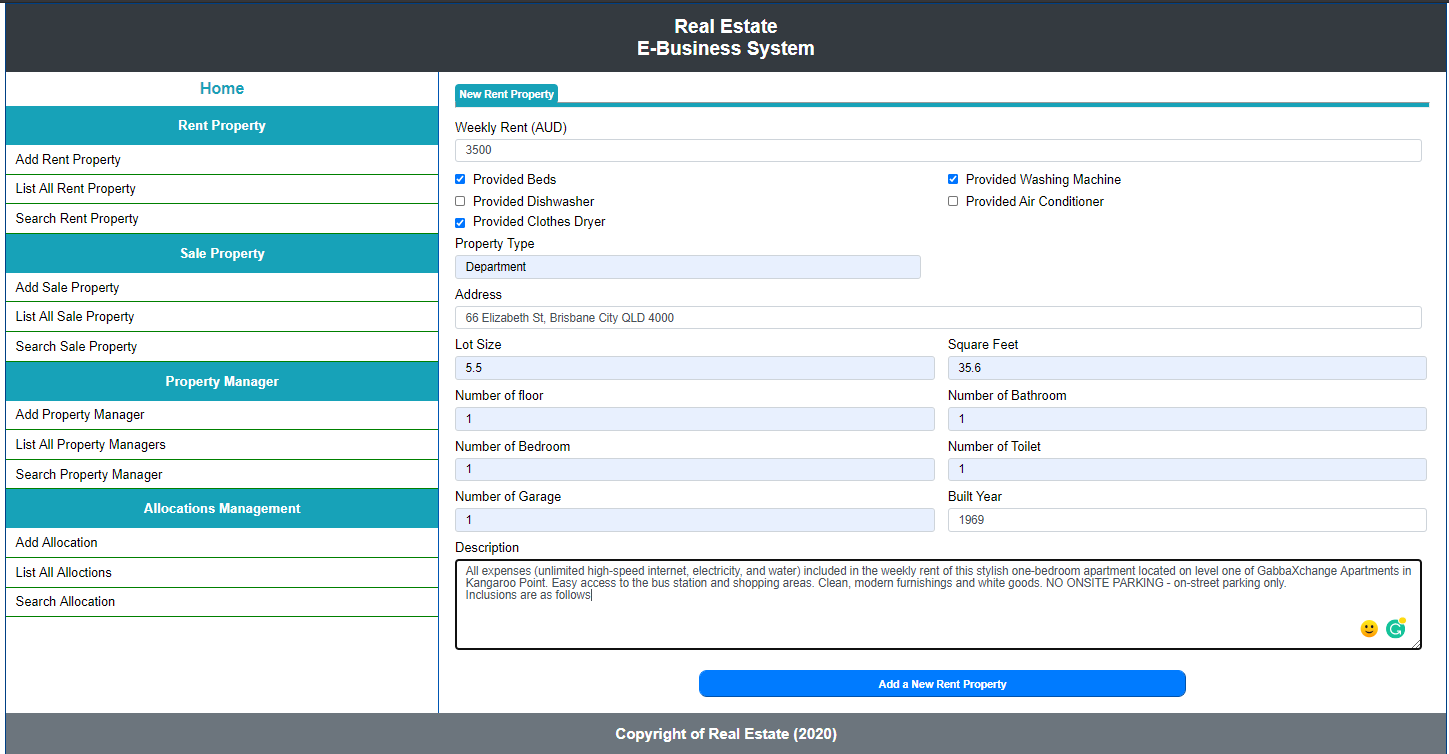


**Save rent property:**

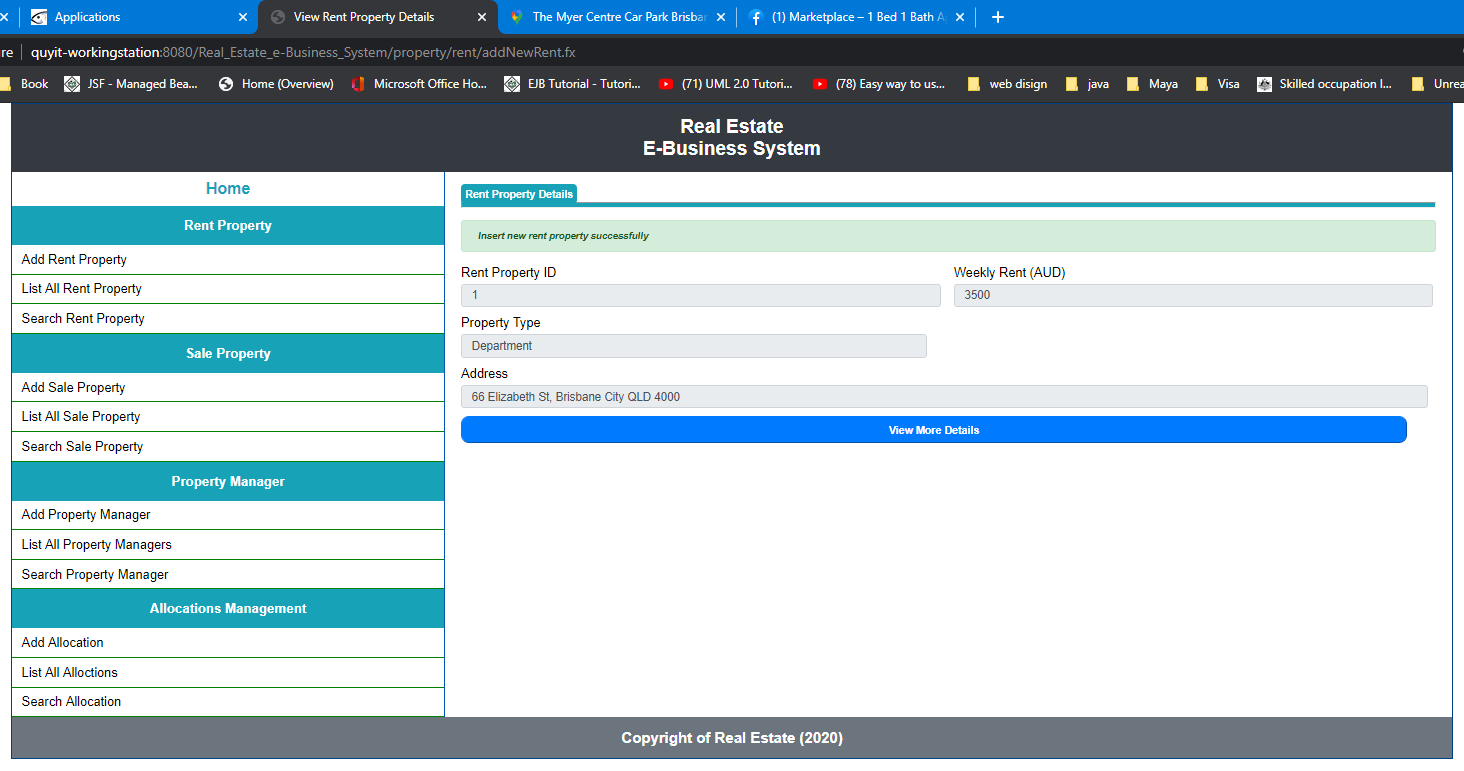
*Incorrect information format:*



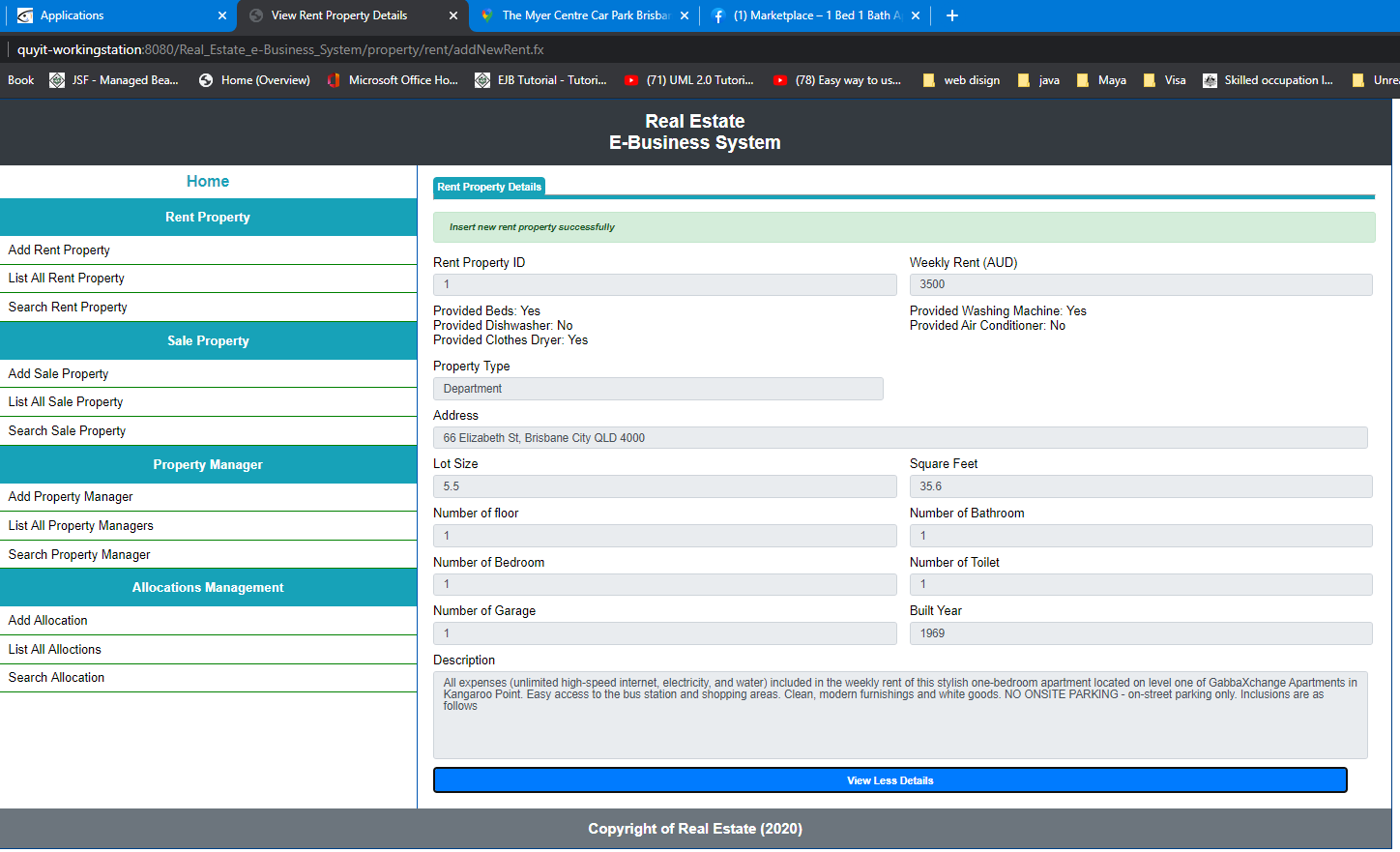
*Correct information format:*



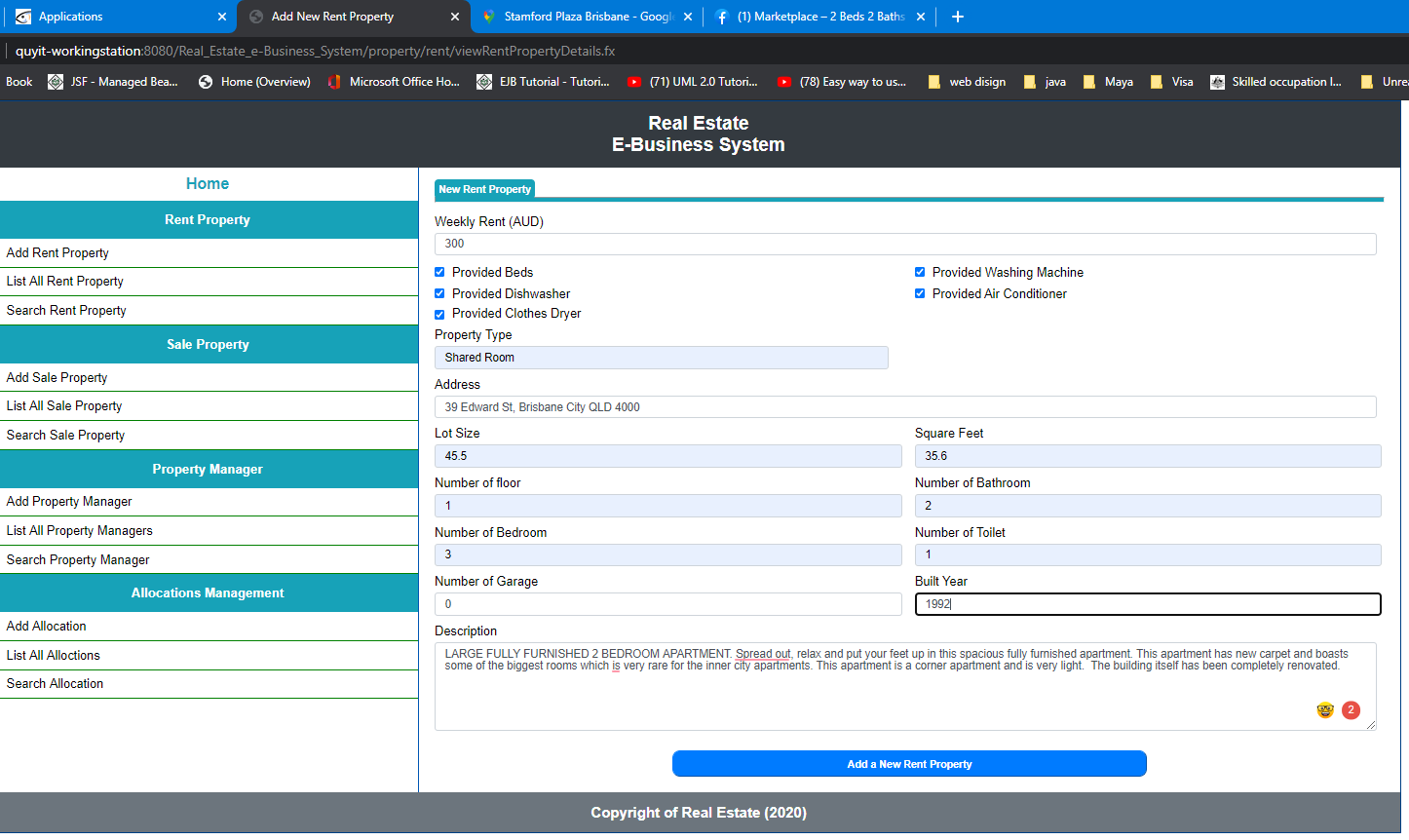
*Result:*



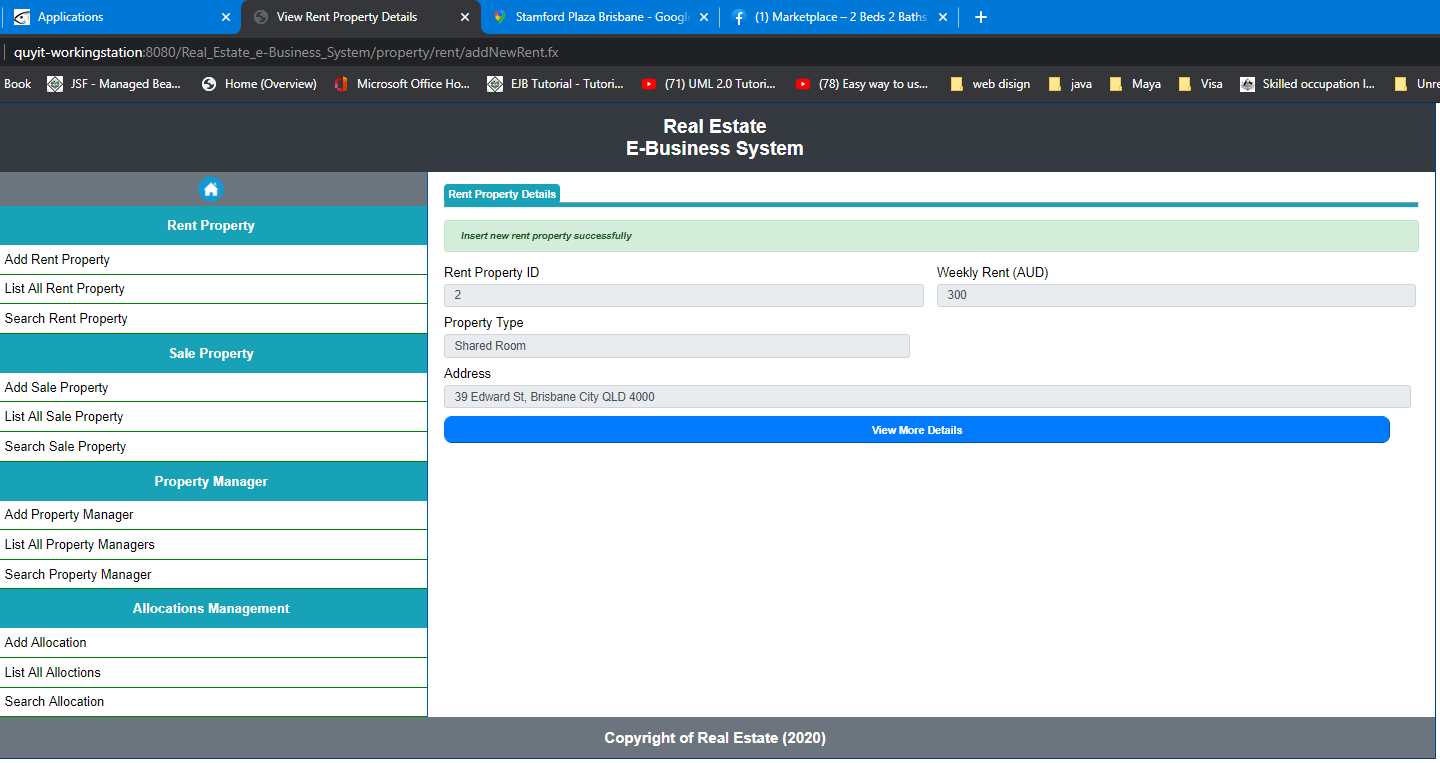
*Click on see more details:*



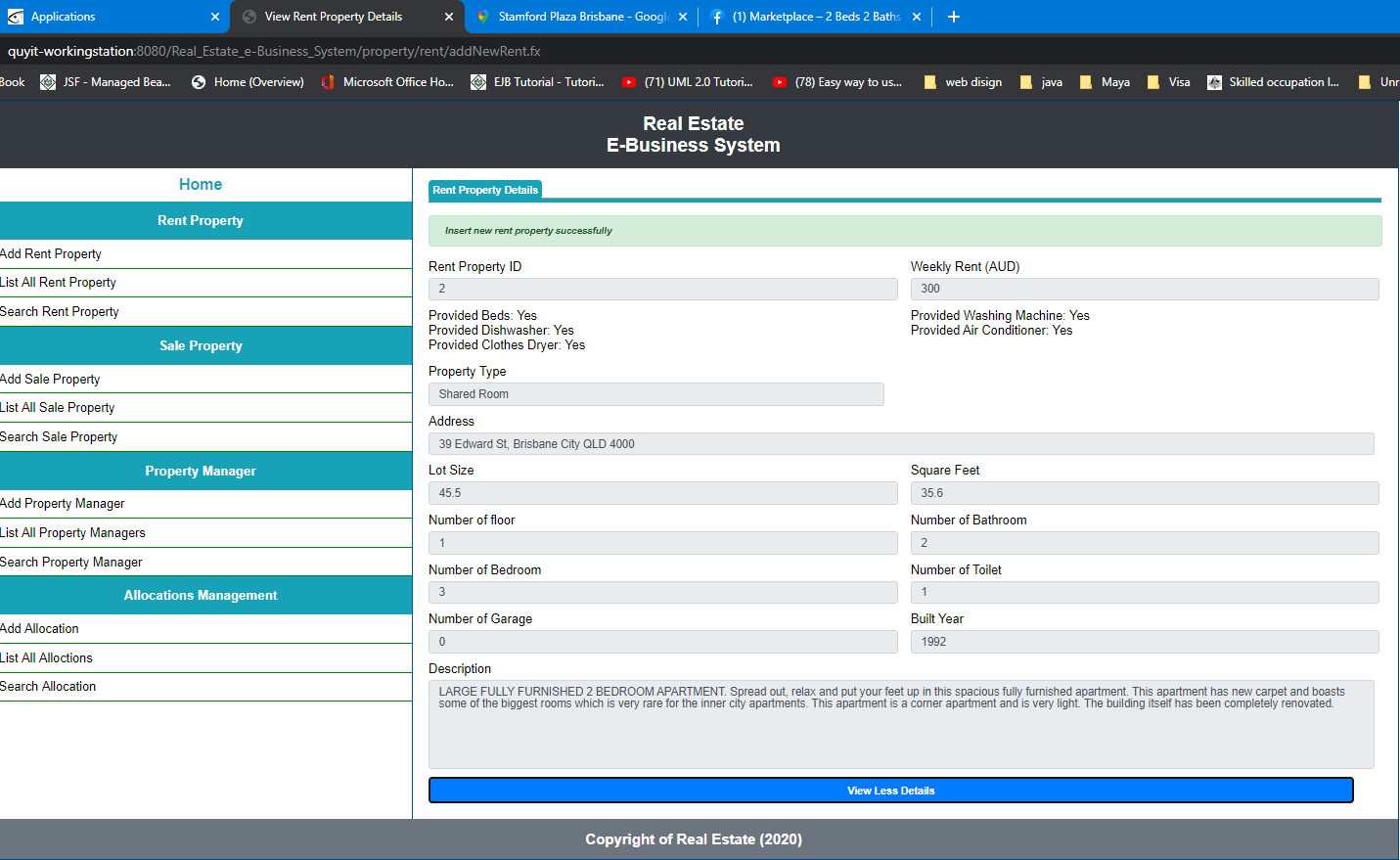
*Save another rent property:*



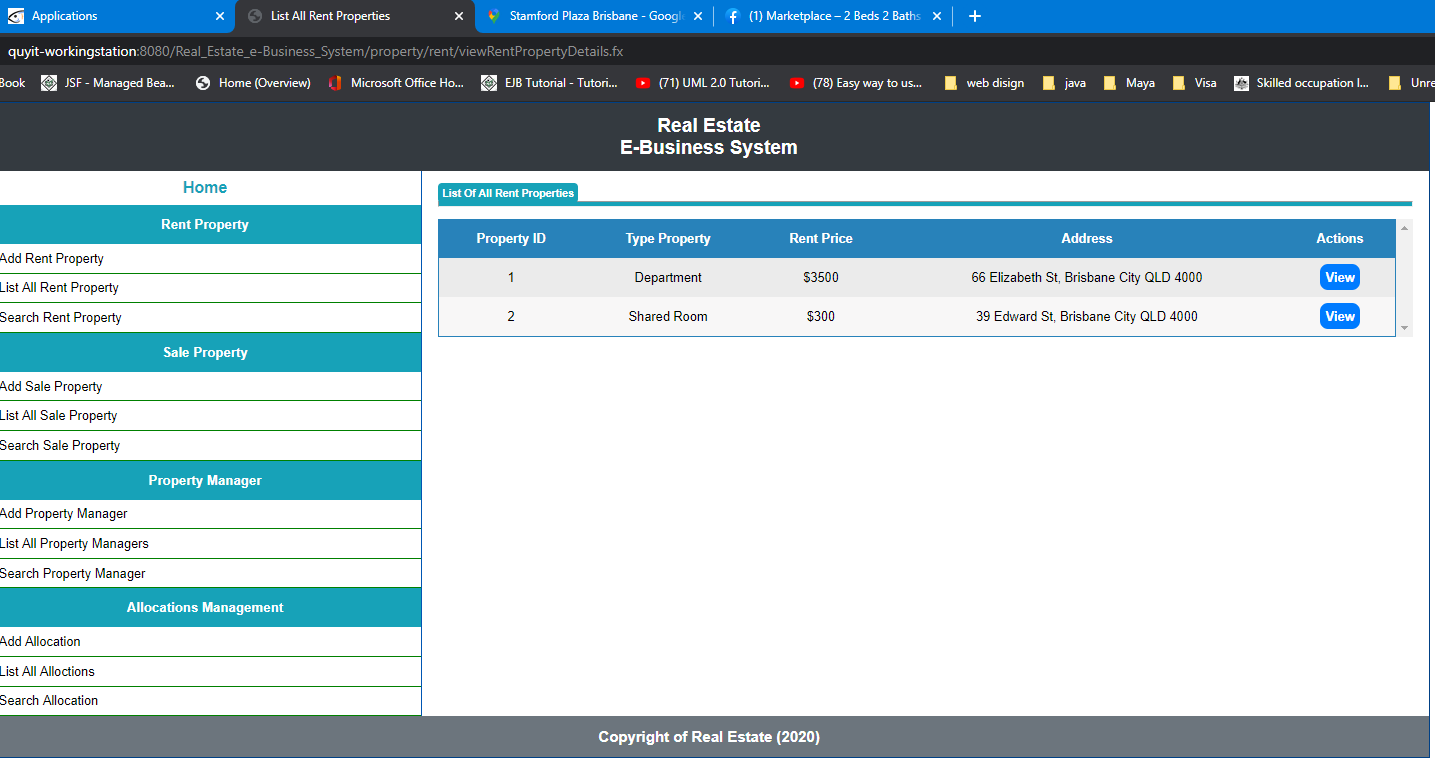
*Successfully saving:*



*Click on view more details:*



***List all rent property:***

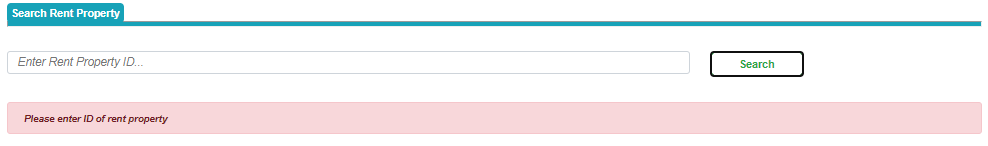


***Click on view:***

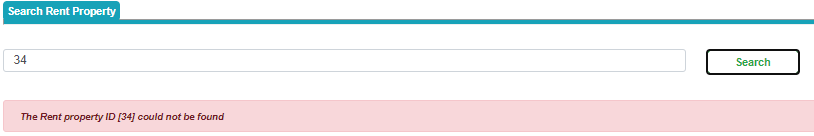


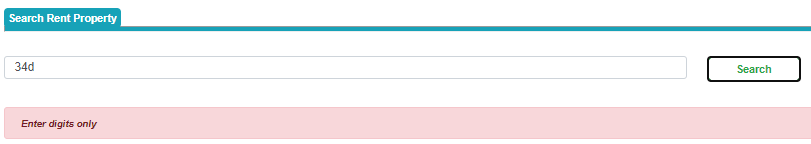
***Search rent property:***

***Empty ID:***

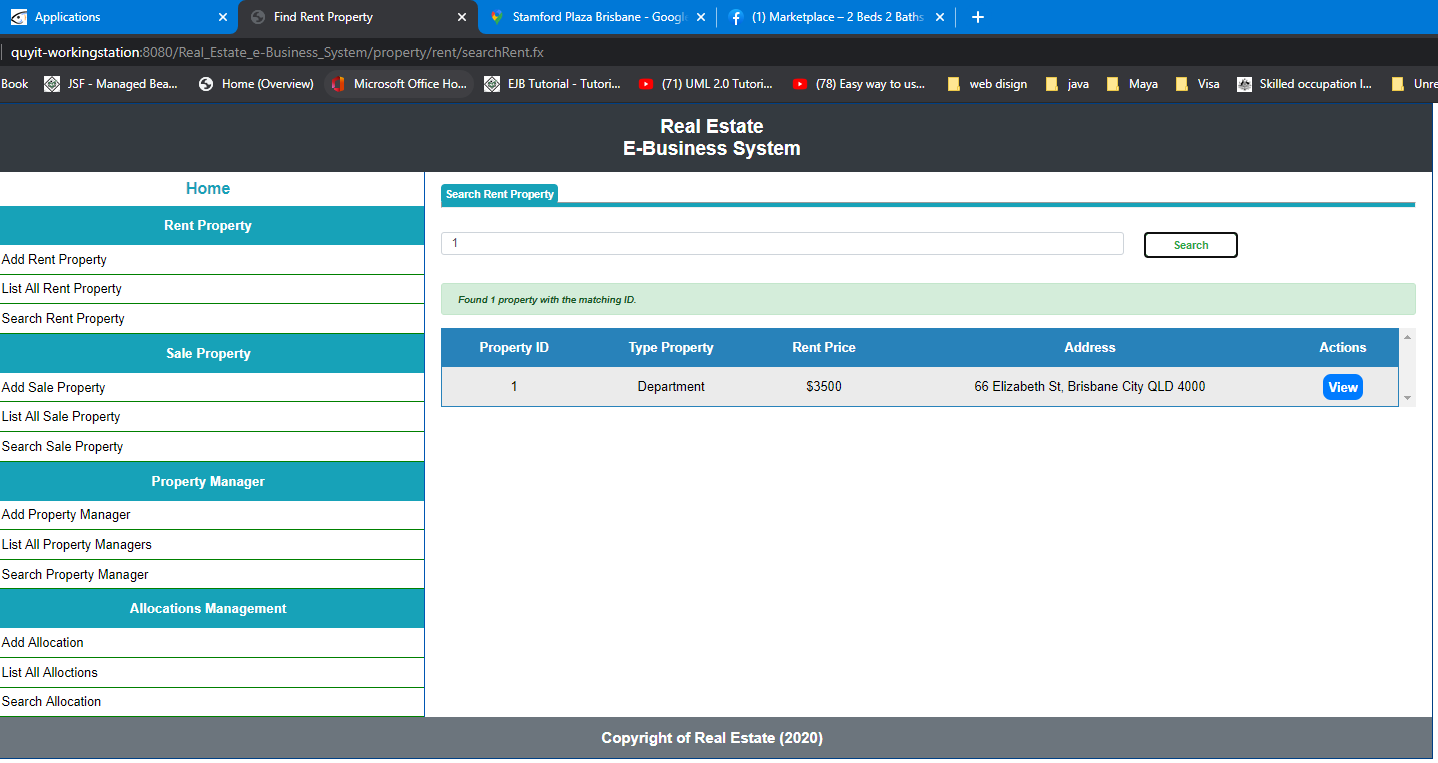


***Enter not existing property ID:***

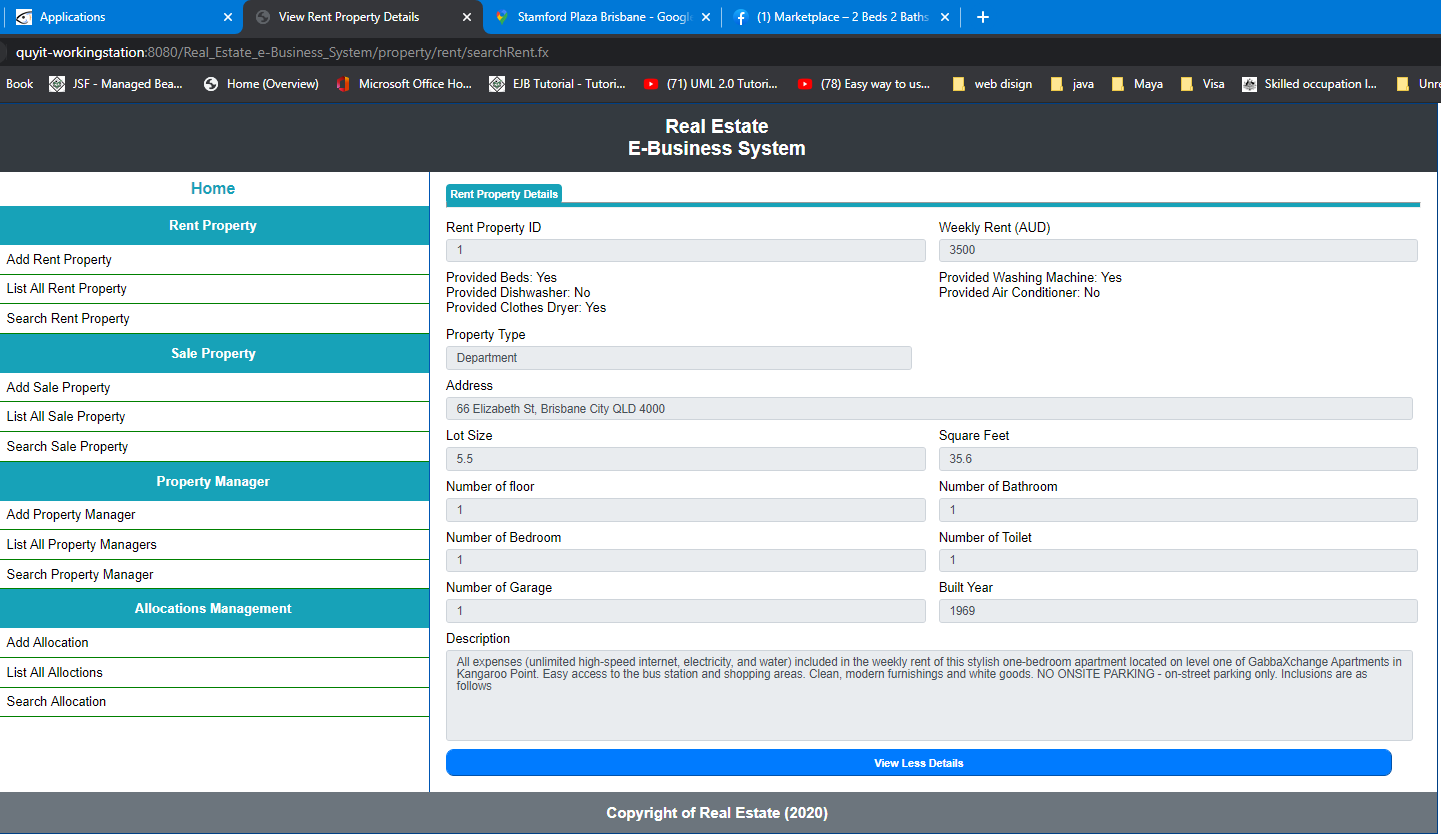
***Enter not digits:***



***Enter existing ID:***

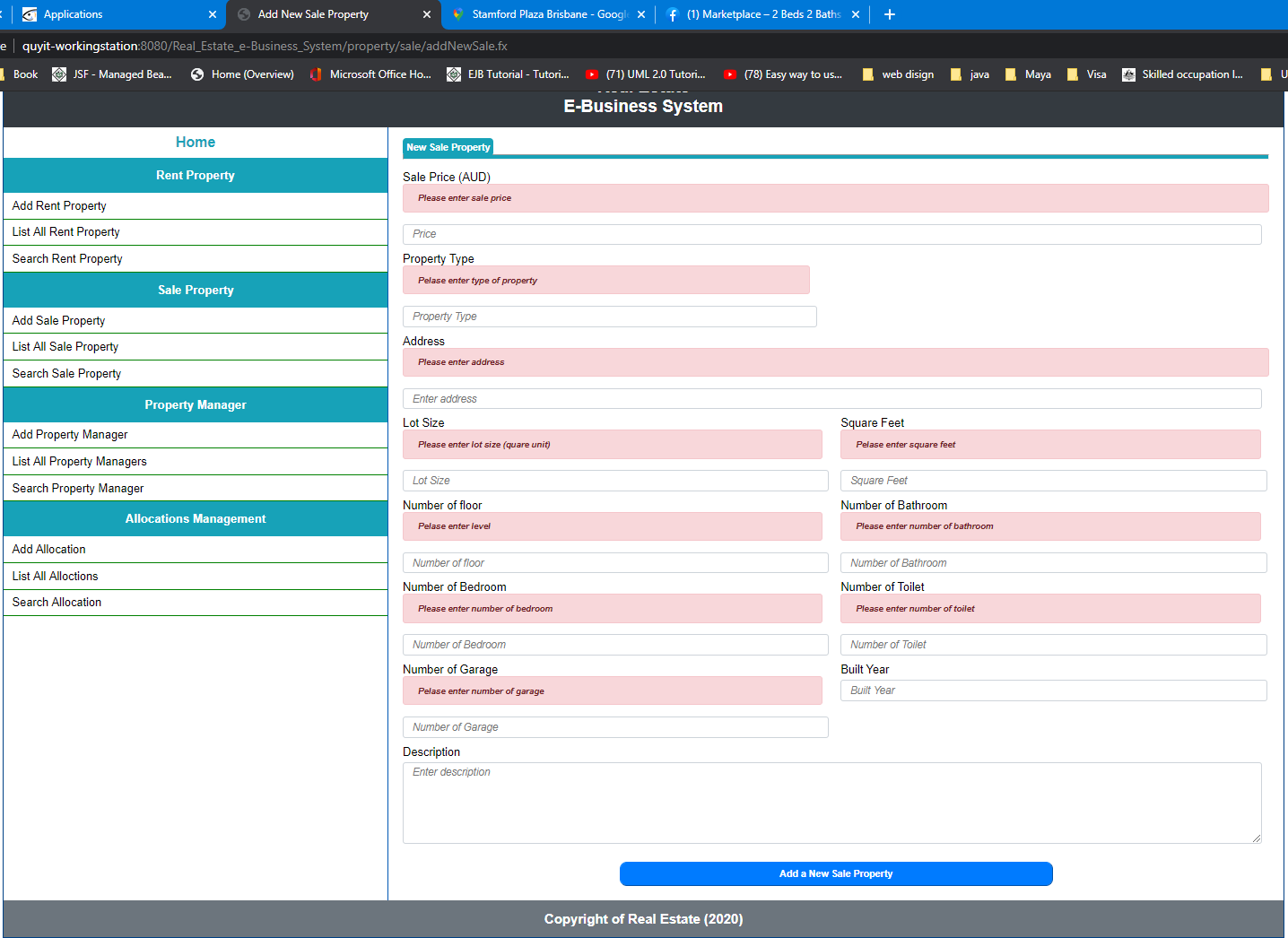


***Click on view:***

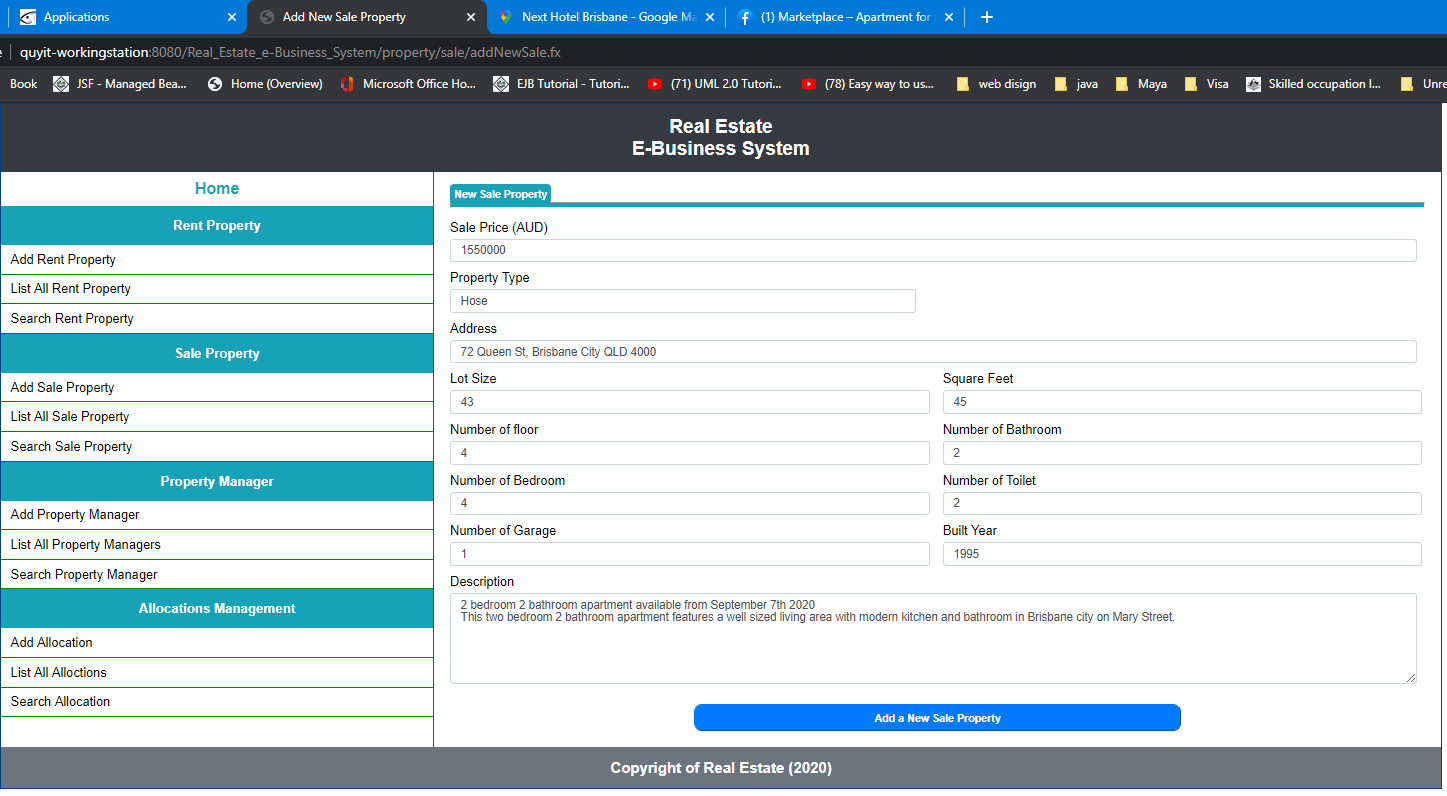


***Save new sale property:***

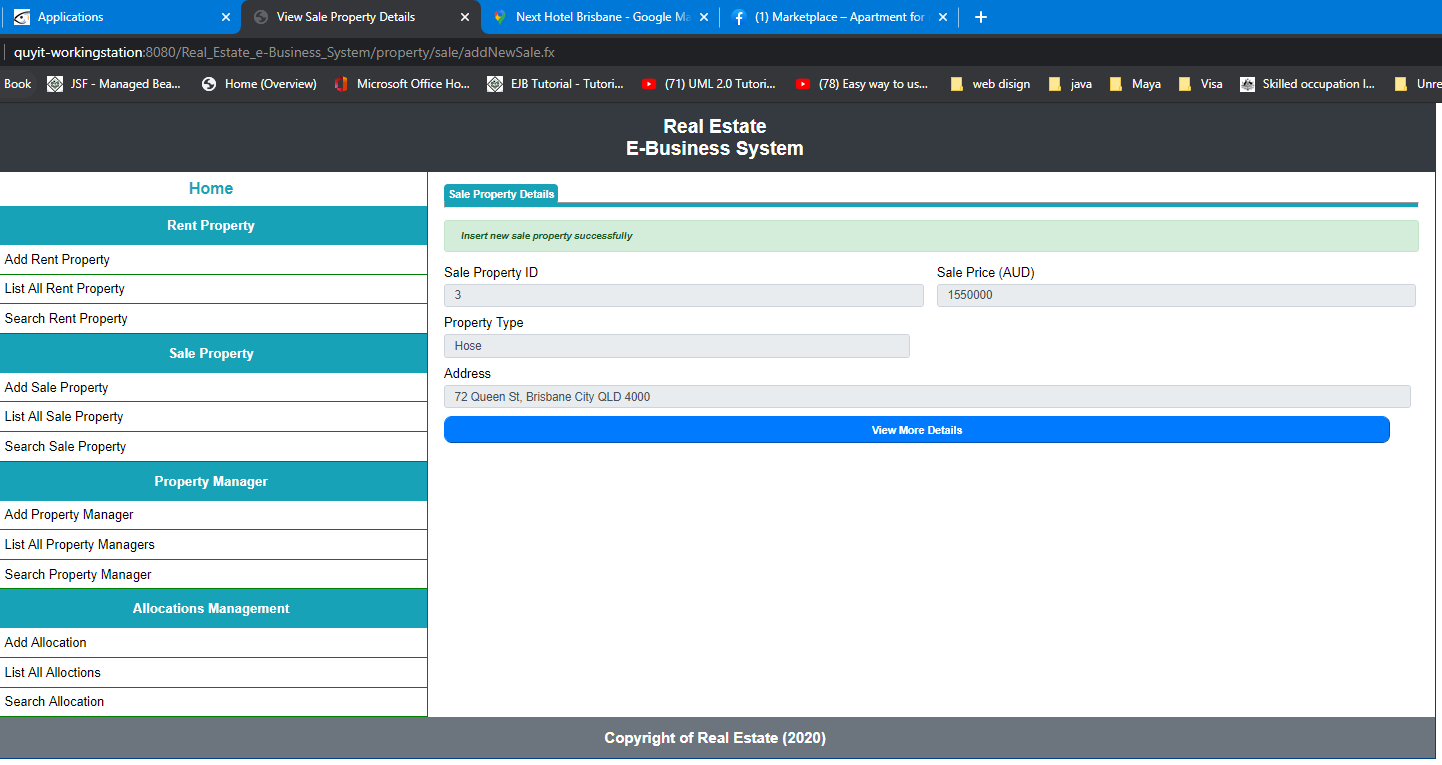
***Empty given information:***



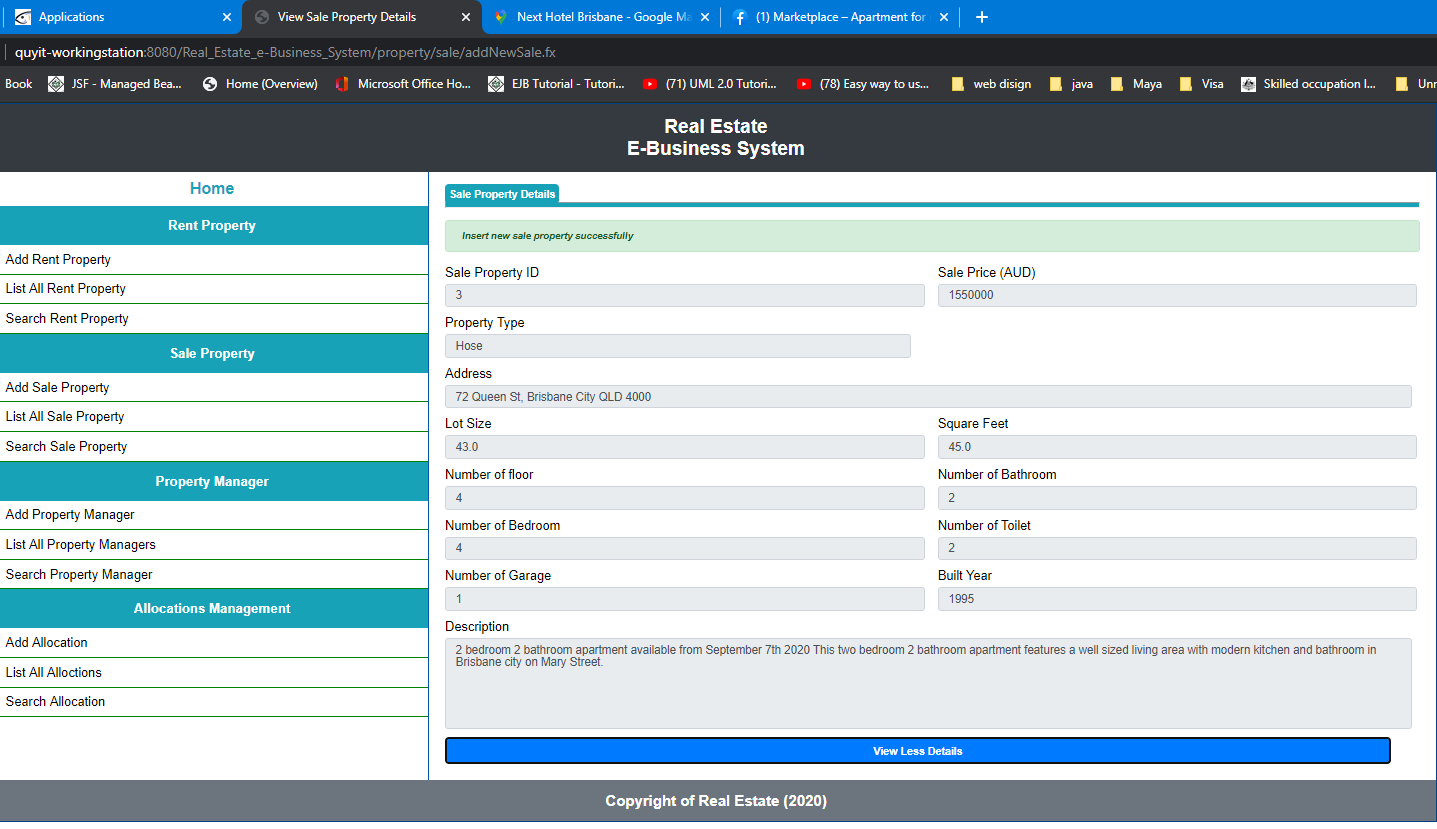
***Correct input:***



***Result:***



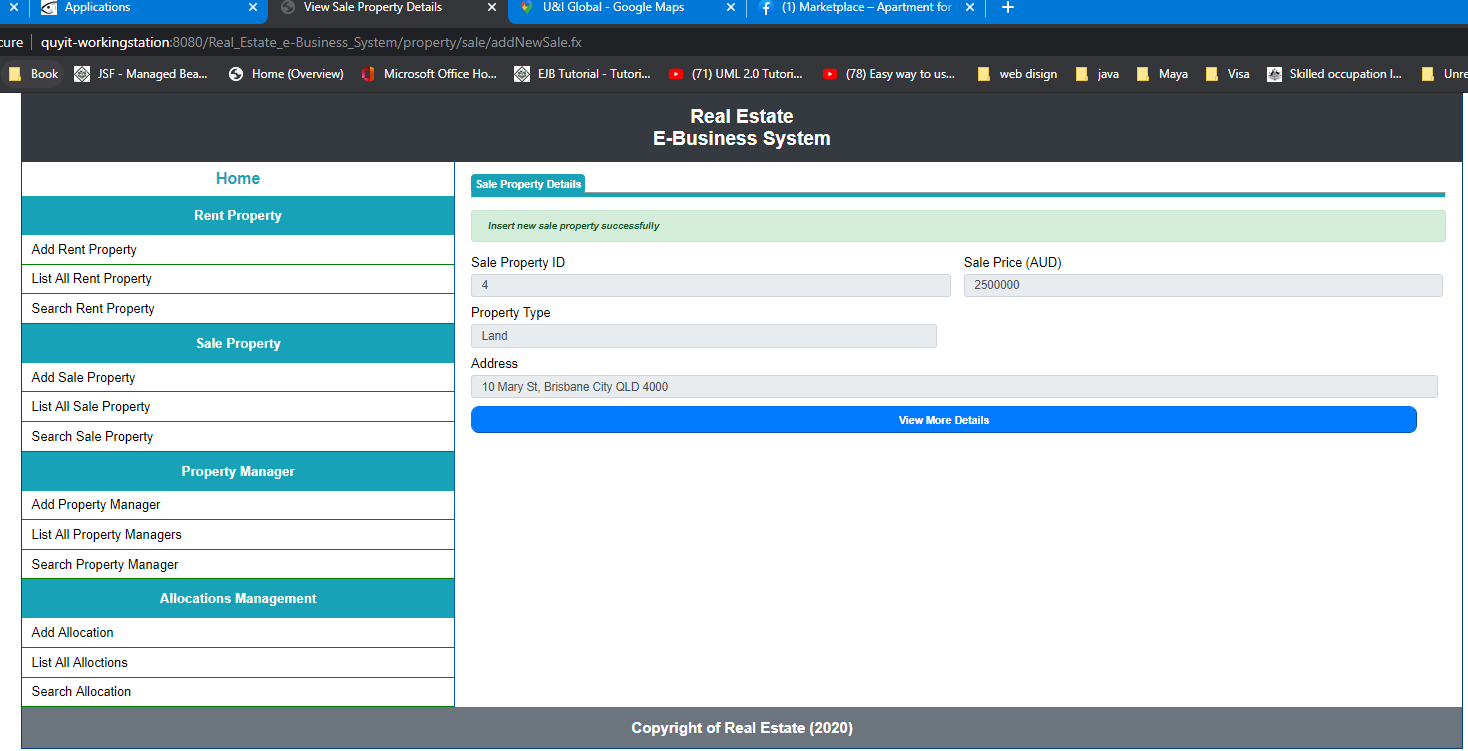
***Click on see more:***



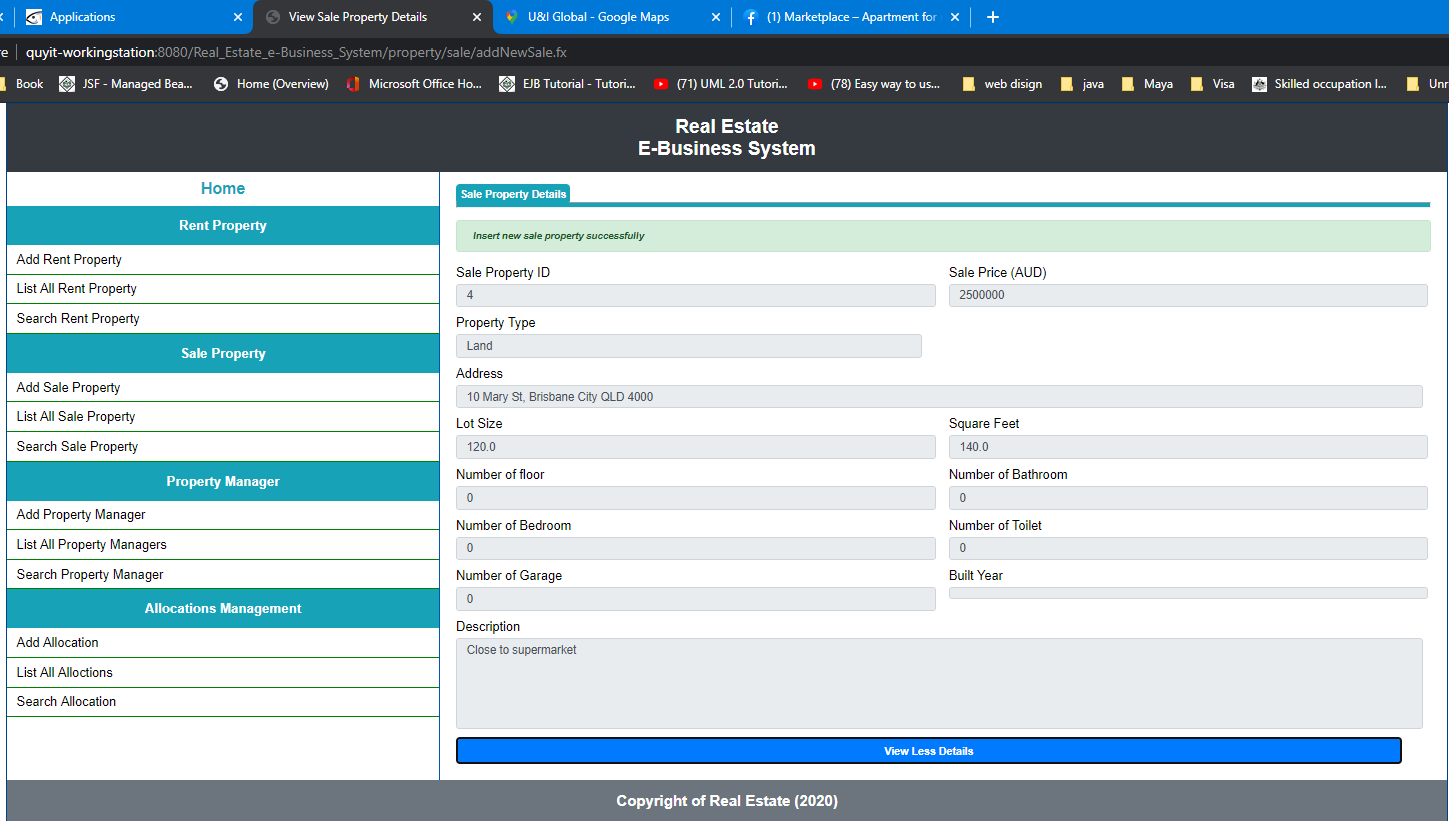
***Add second sale property:***



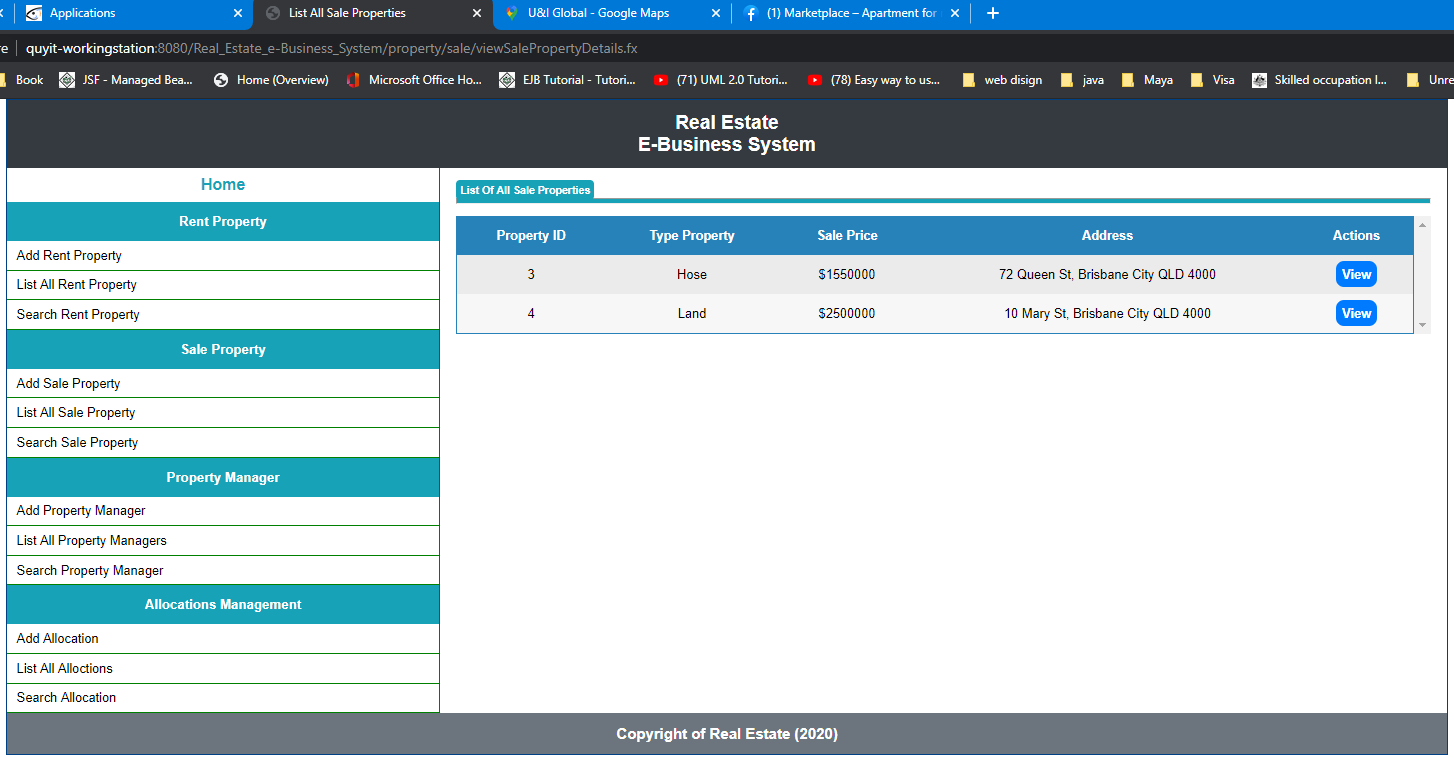
***Result:***



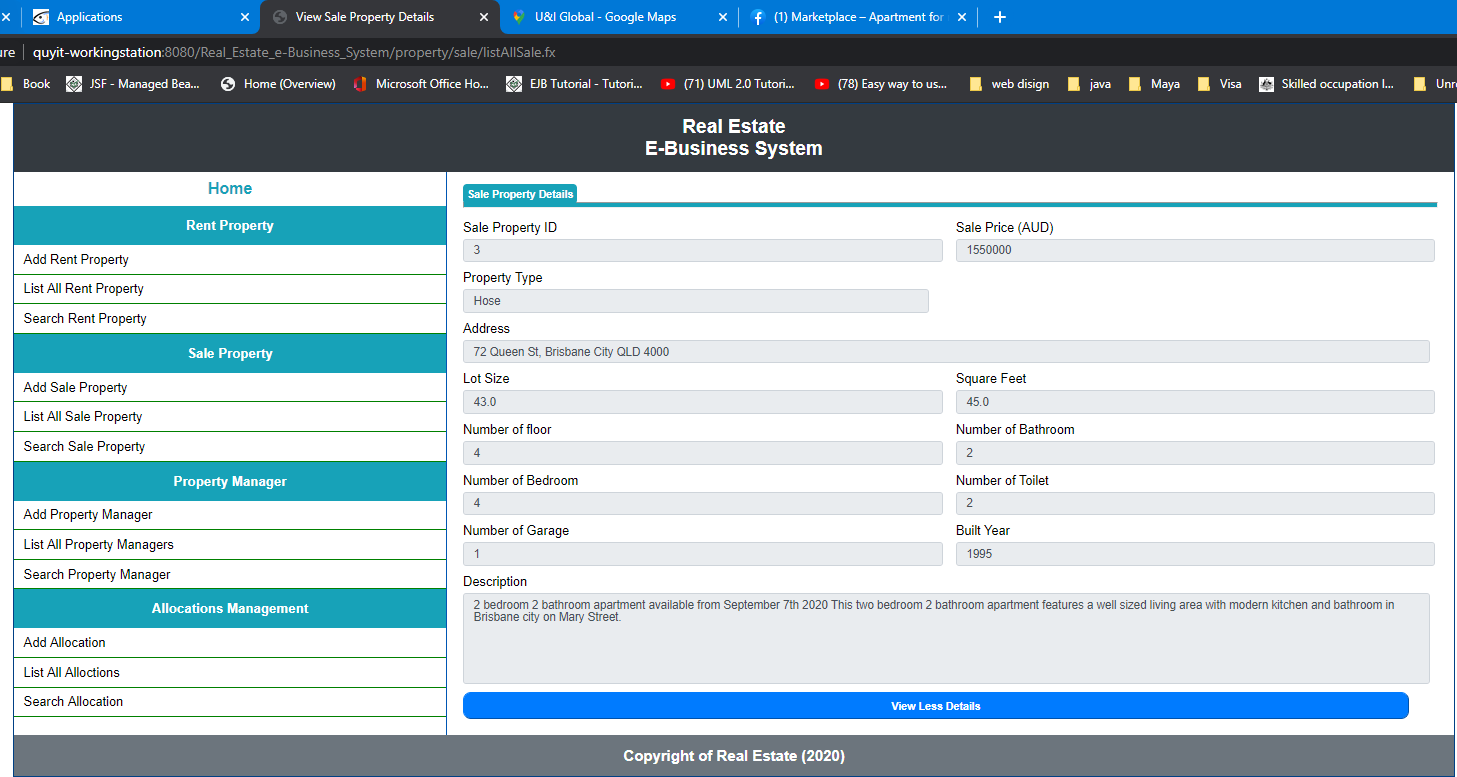
***Click on view more details:***



***List all sale property:***

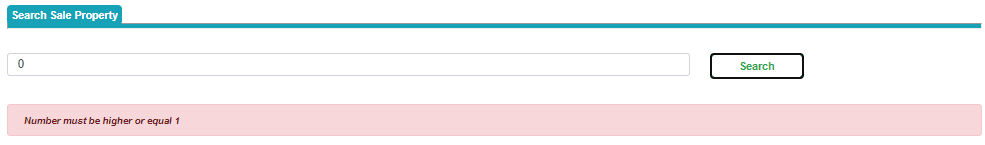


***Click on view:***

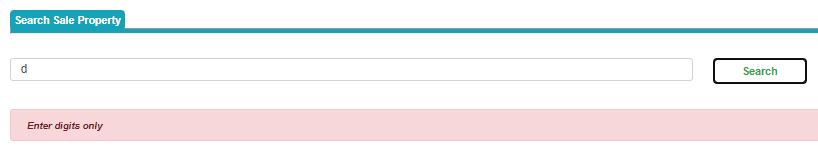


***Search sale property:***

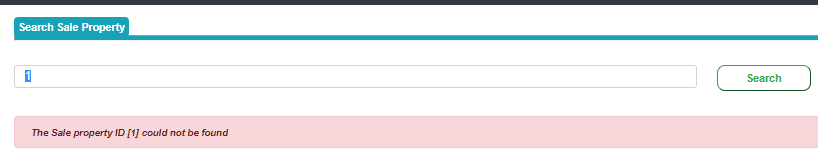
***Empty ID:***



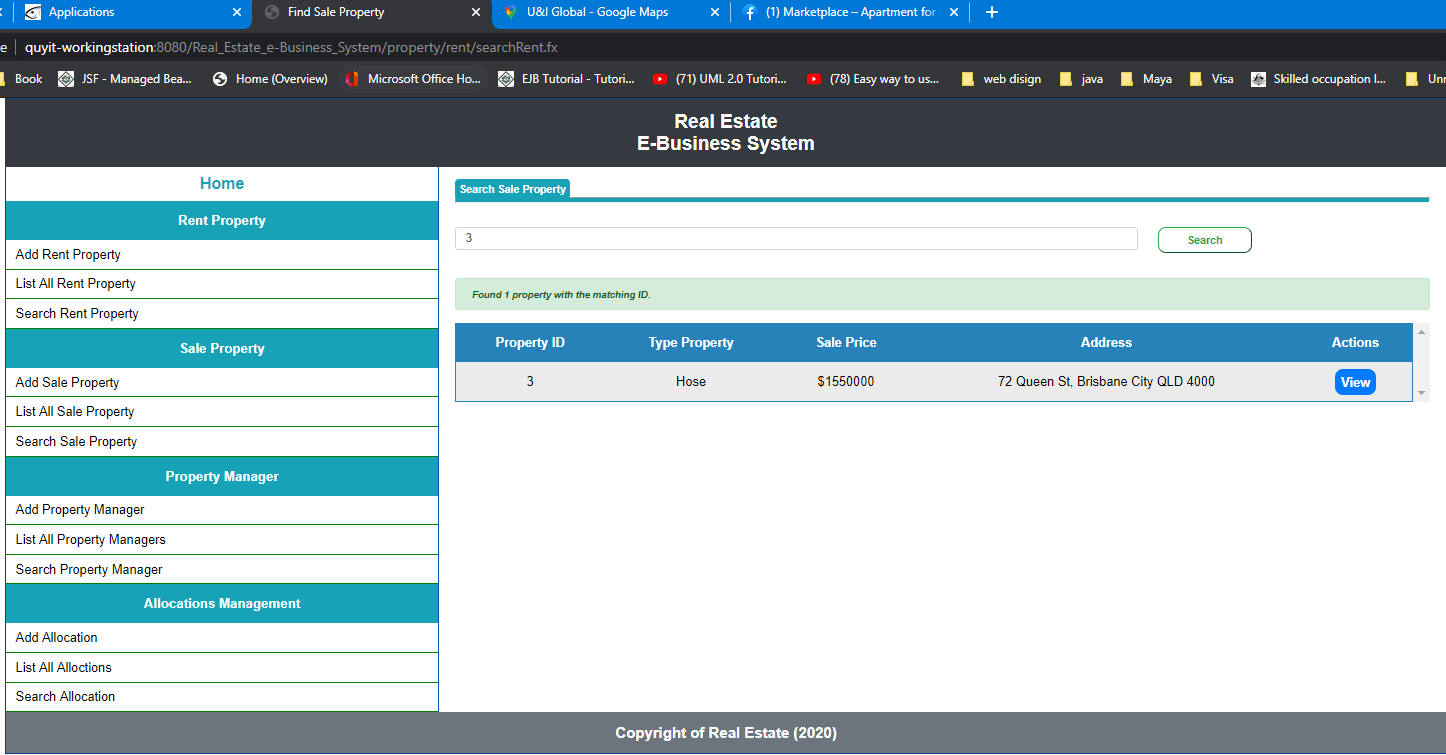
***Not enter number:***



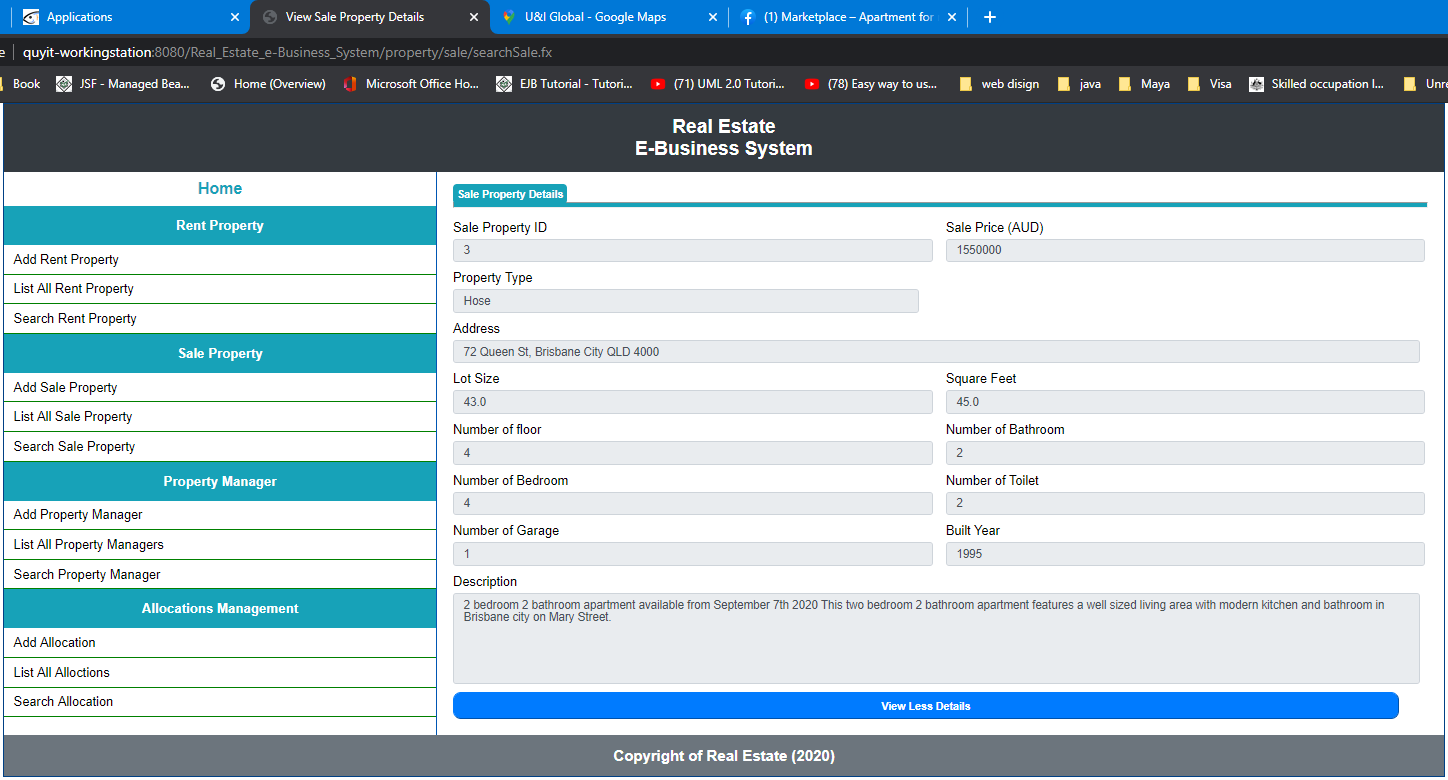
***Not existing ID:***



***Existing ID:***

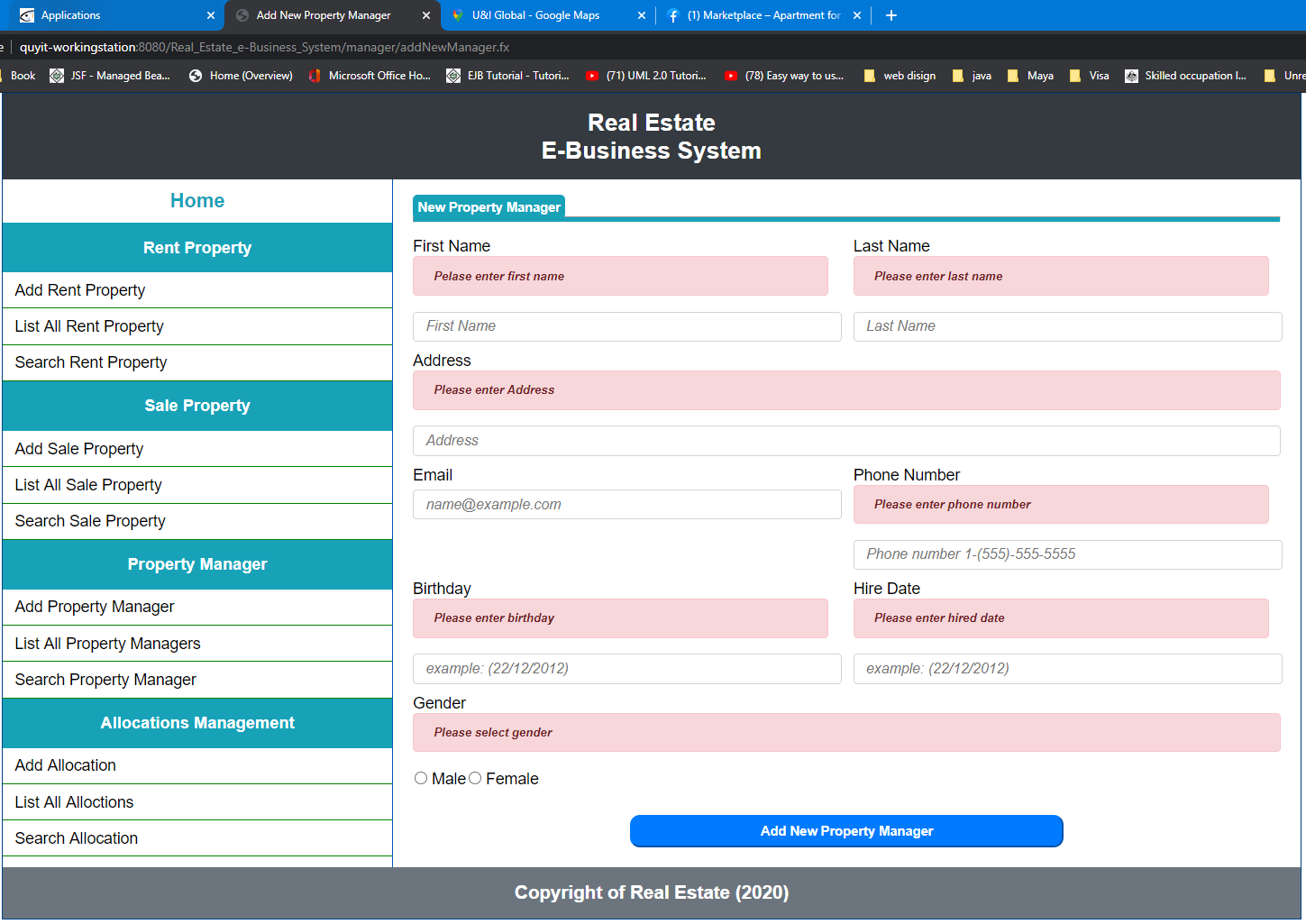


***Click on view:***

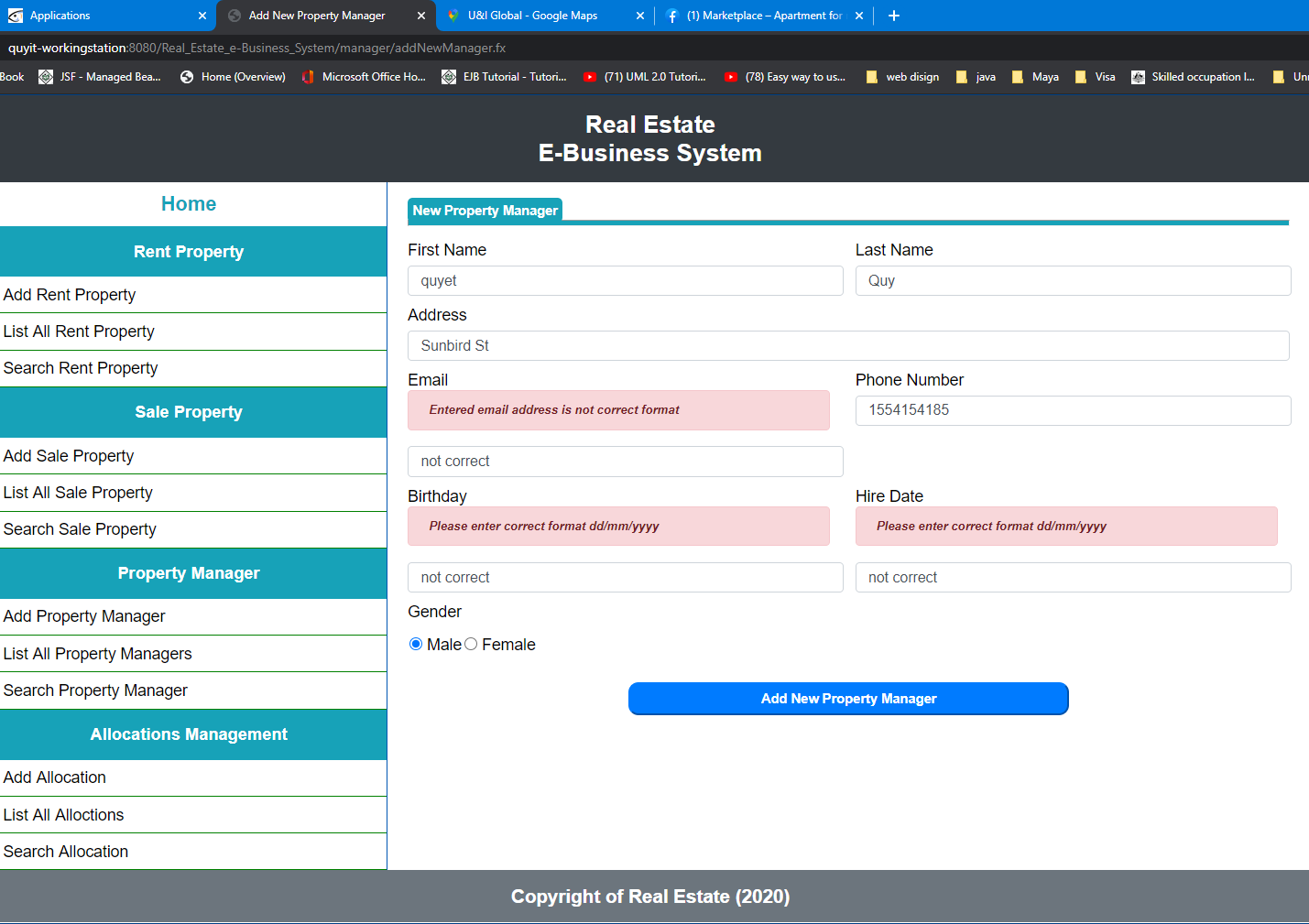


***Save new property manager:***

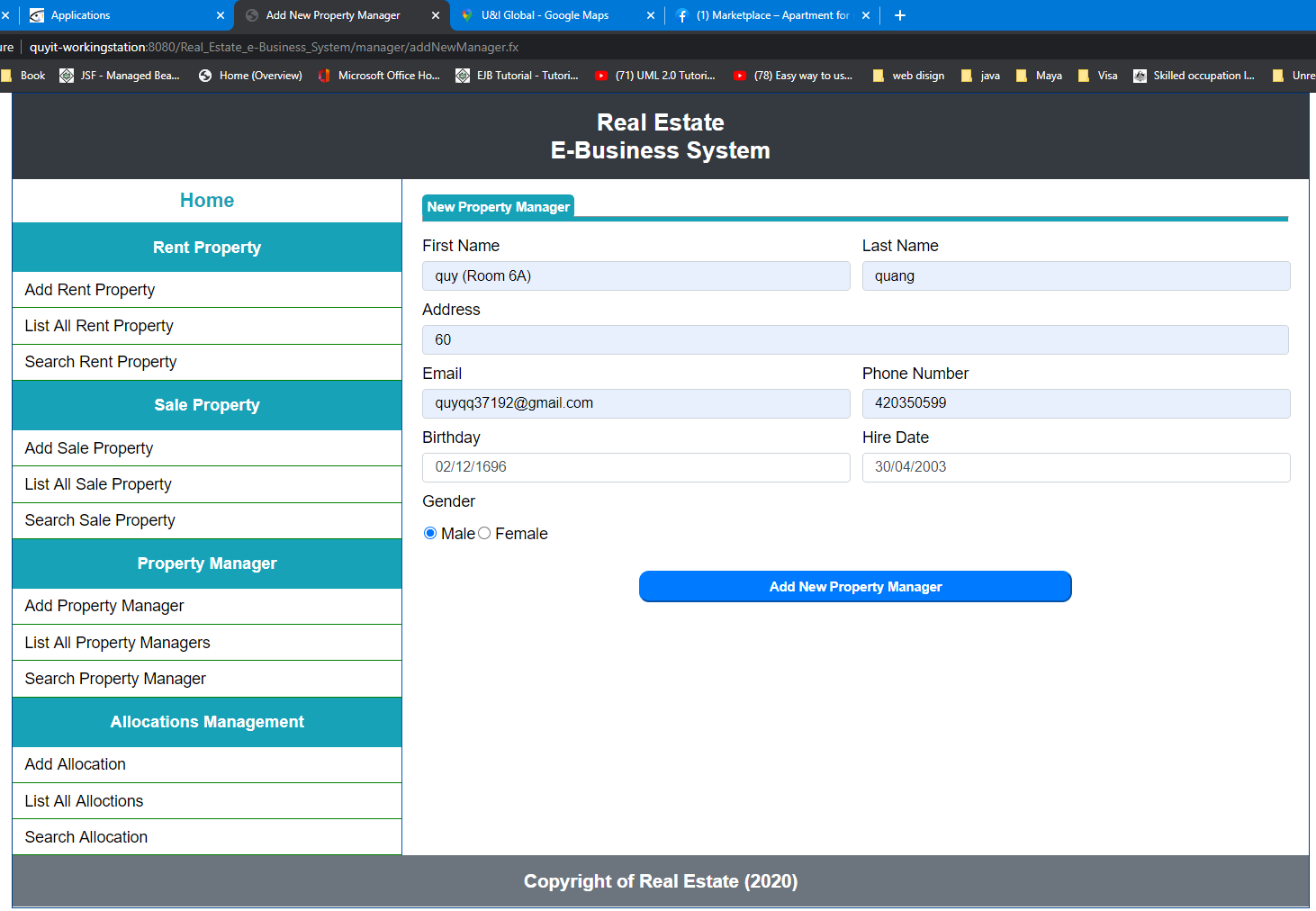
***Empty input:***



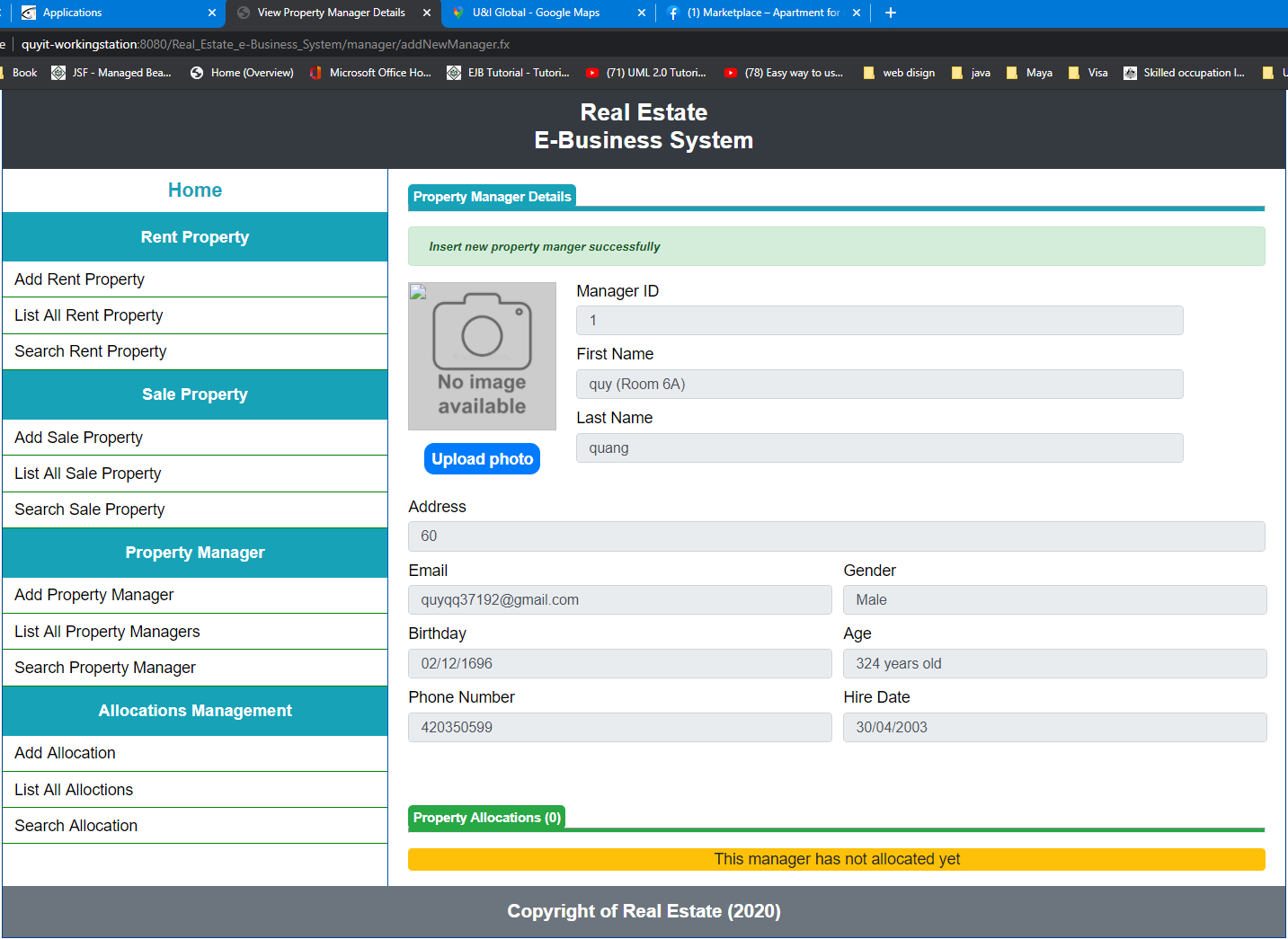
***Incorrect format of input:***



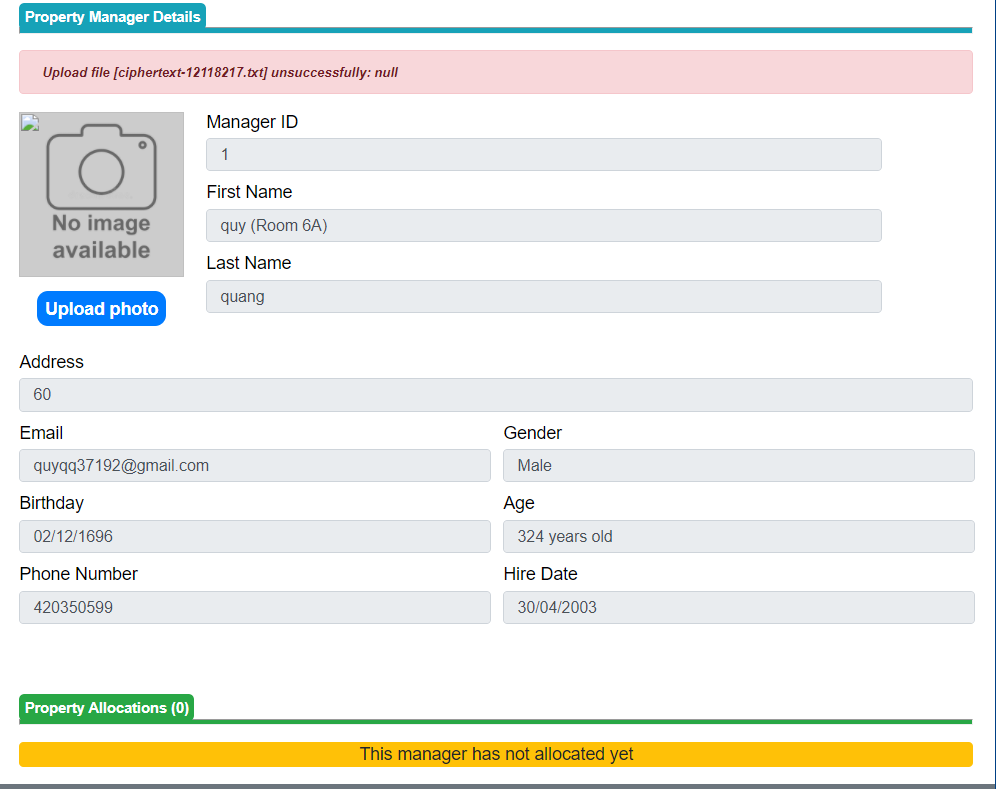
***Correct input:***



***Result:***



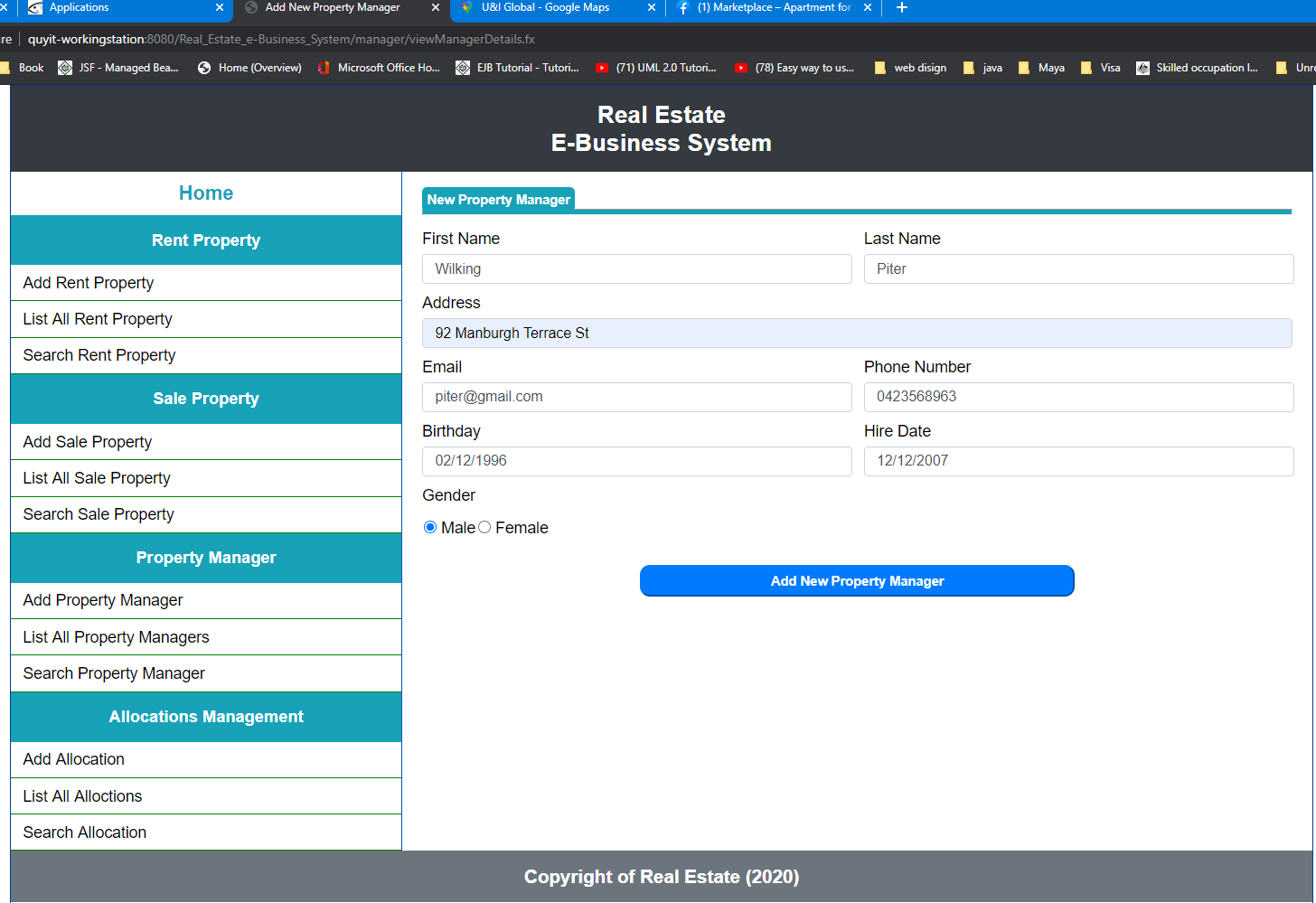
***Upload photo with unsupported file:***



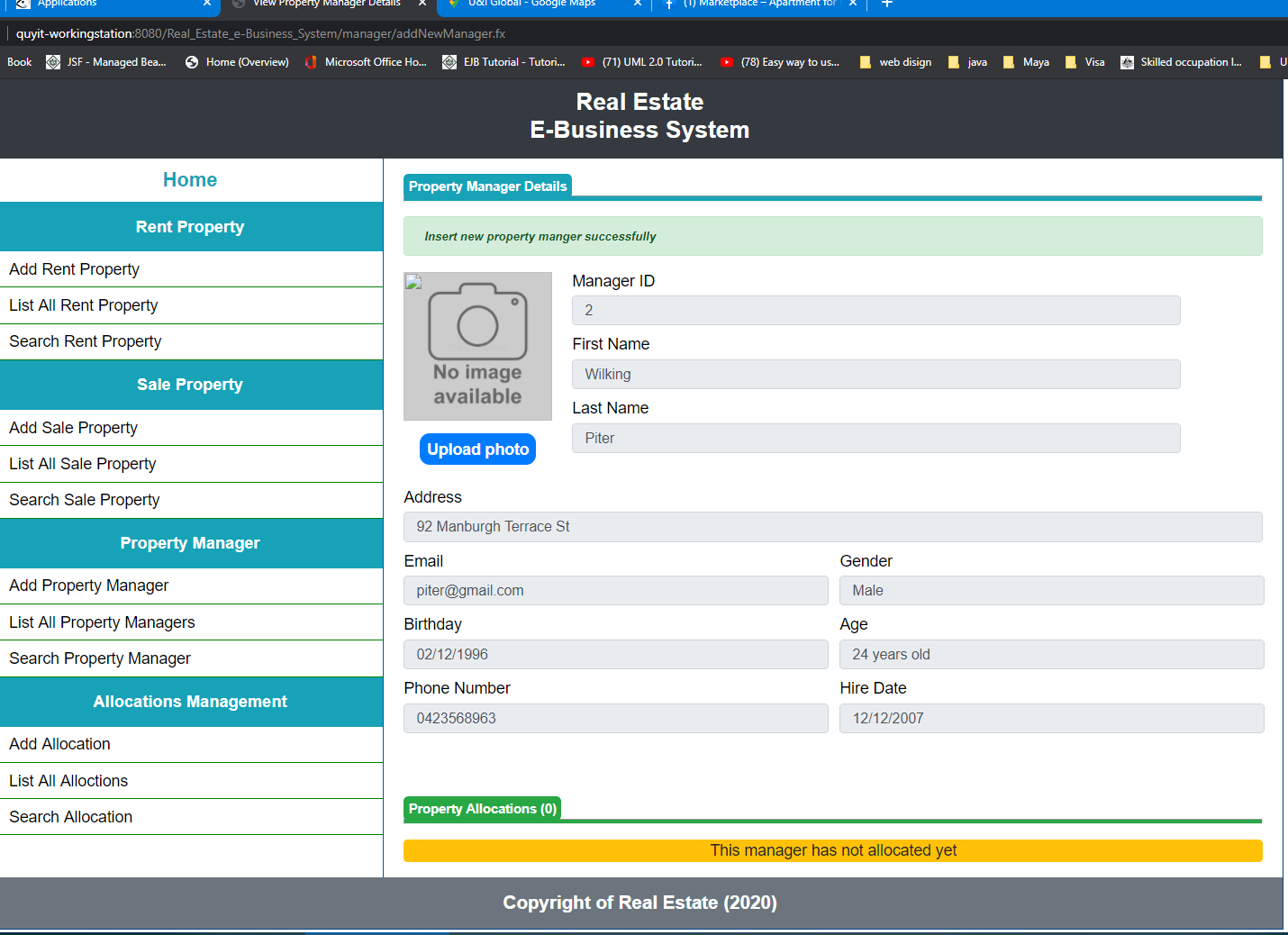
***Upload photo successfully:***



***Add second property manager:***

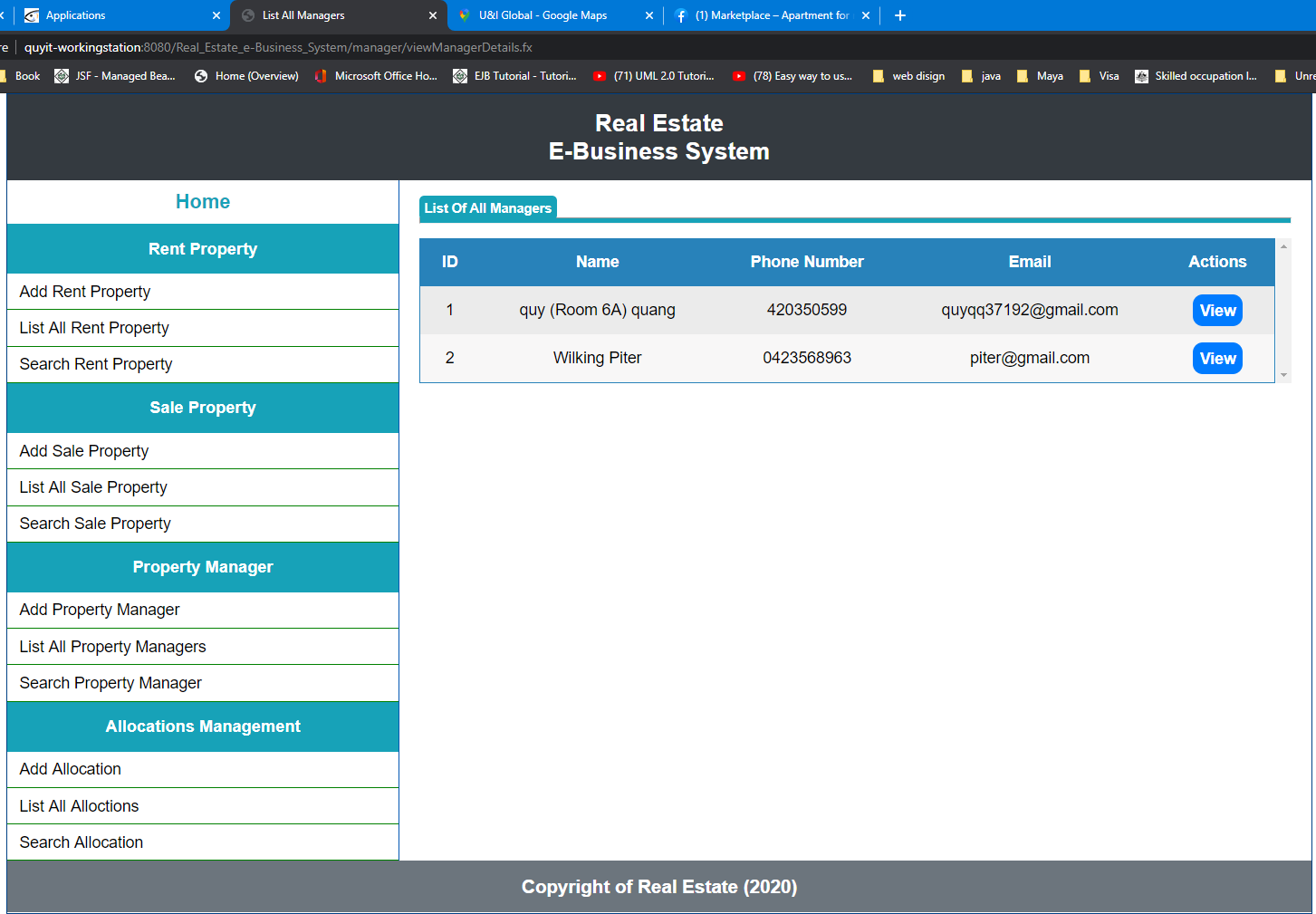


***Result:***

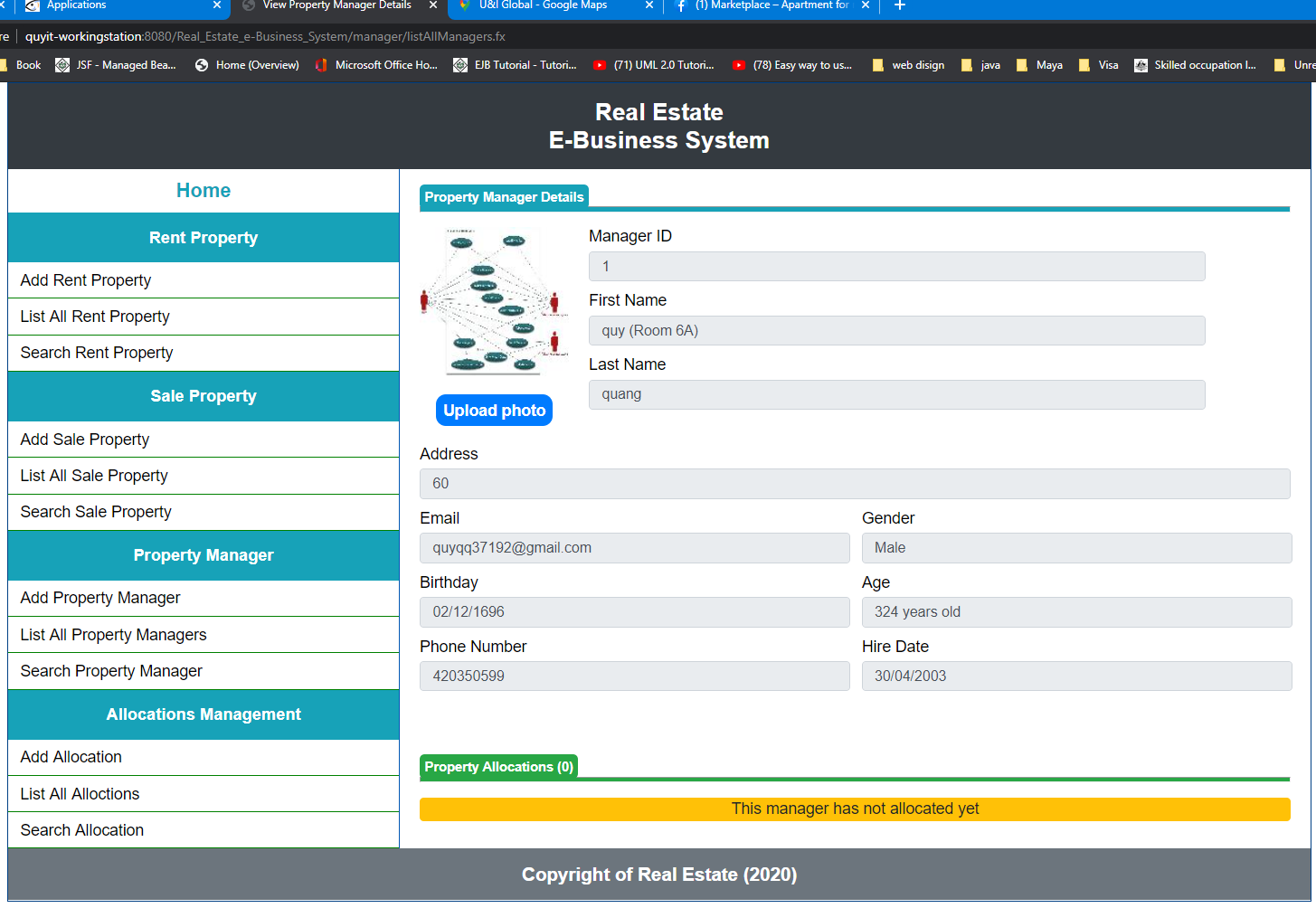


***Not uploading photo.***

***List all property managers:***

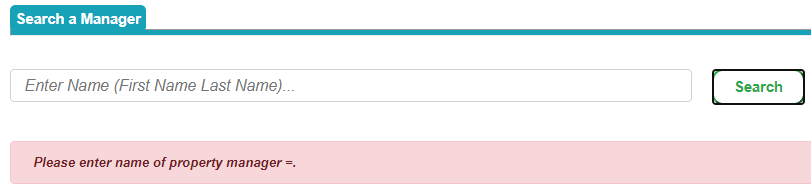


***Click on view:***



***Search property manager:***

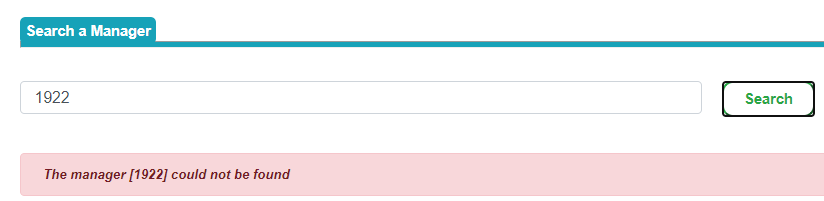
***Empty name:***



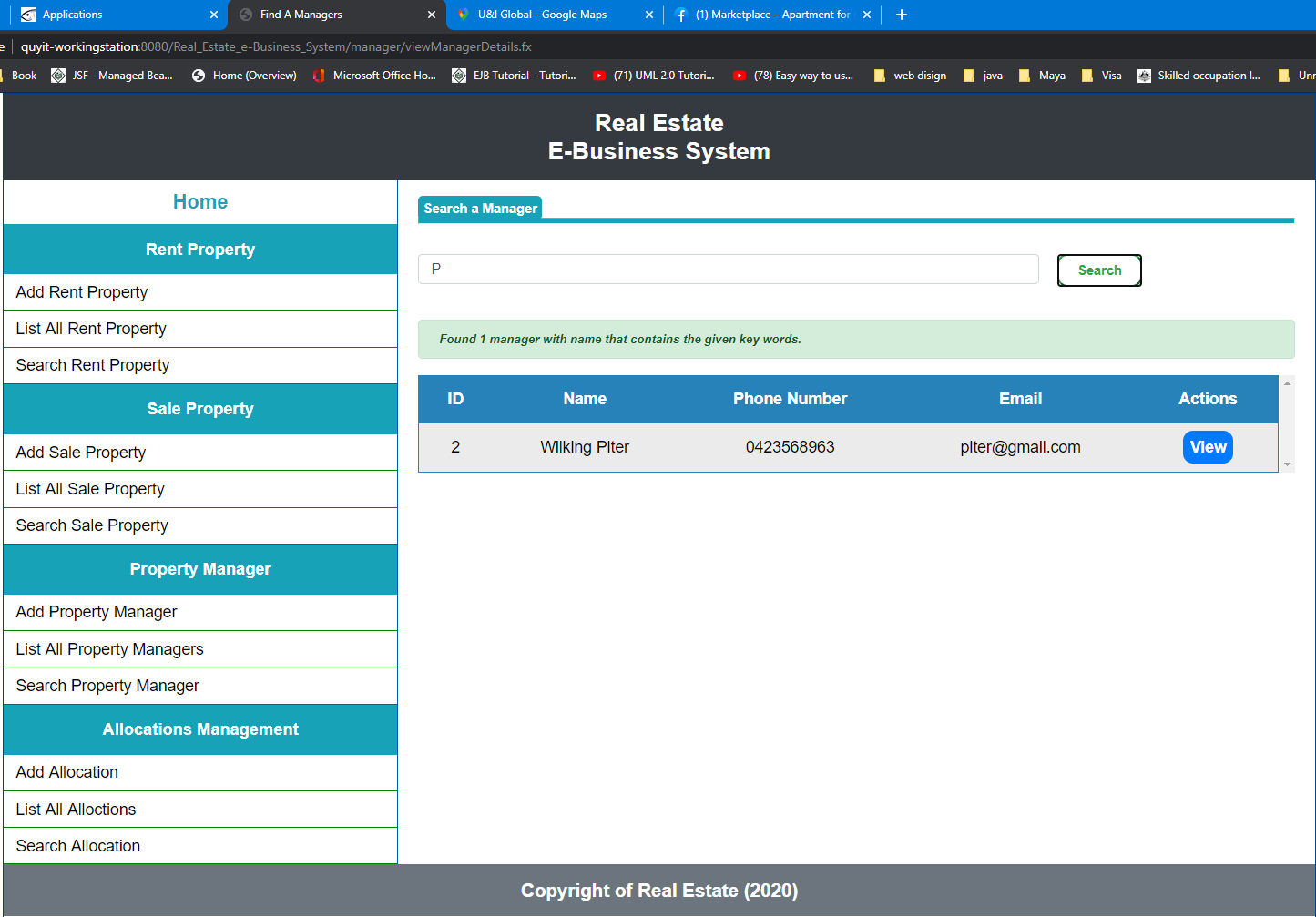
With one letter, this feature is to search all property manager who has name contain keyword:

For example: keyword is “A” it will find all property manager who has name contain letter “A”

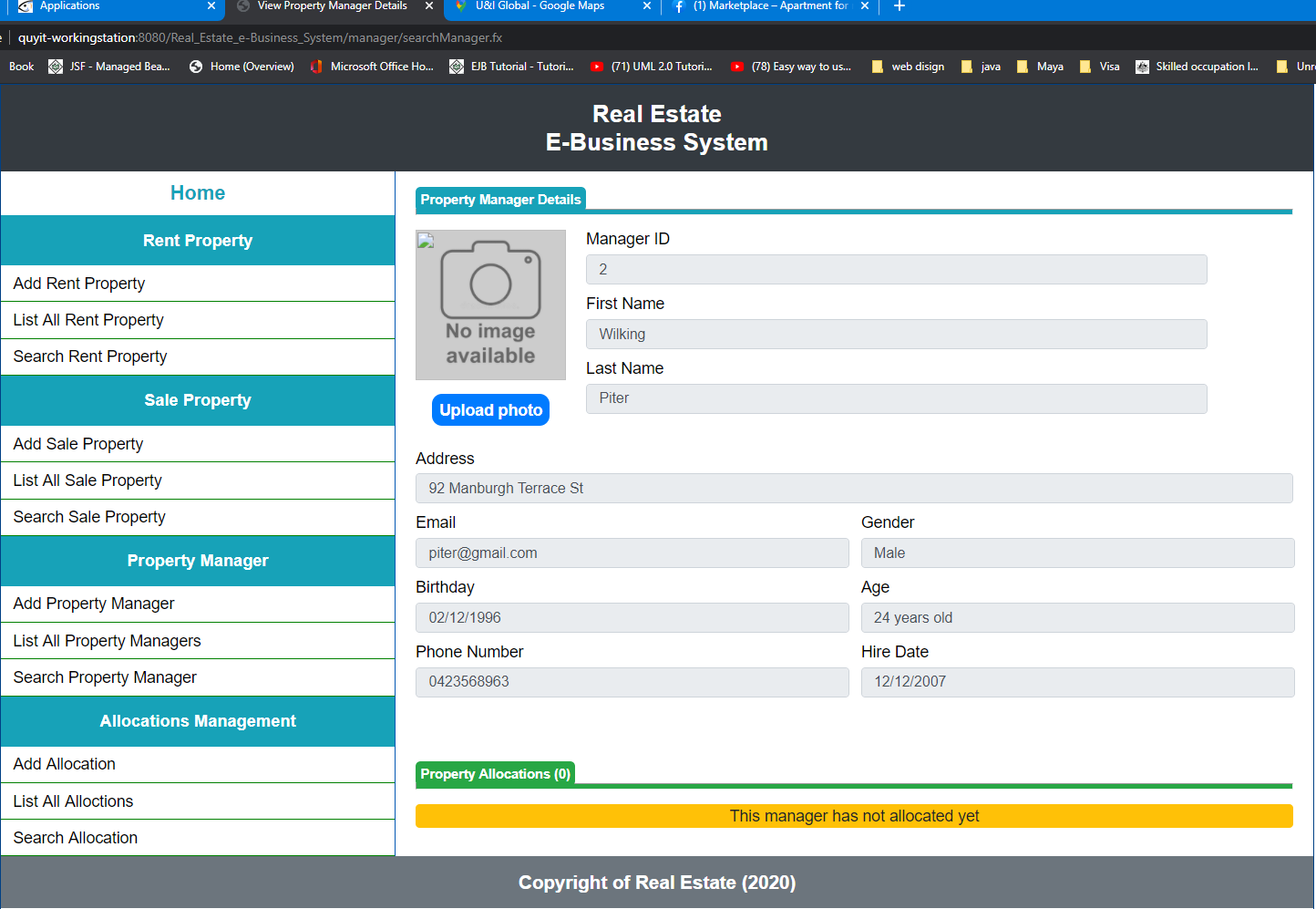
***Not existing name:***



***Matching name:***

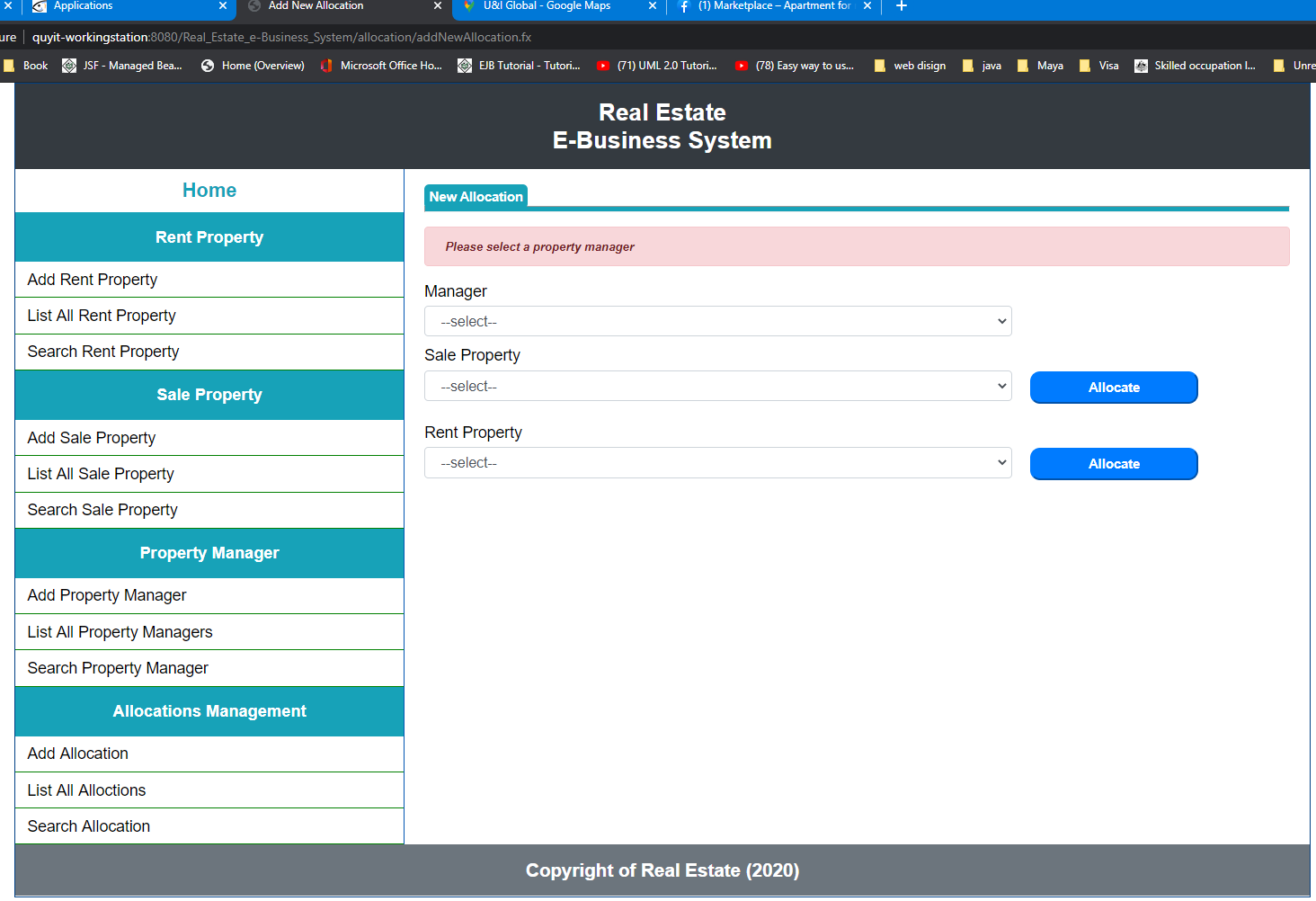


***Click on view:***

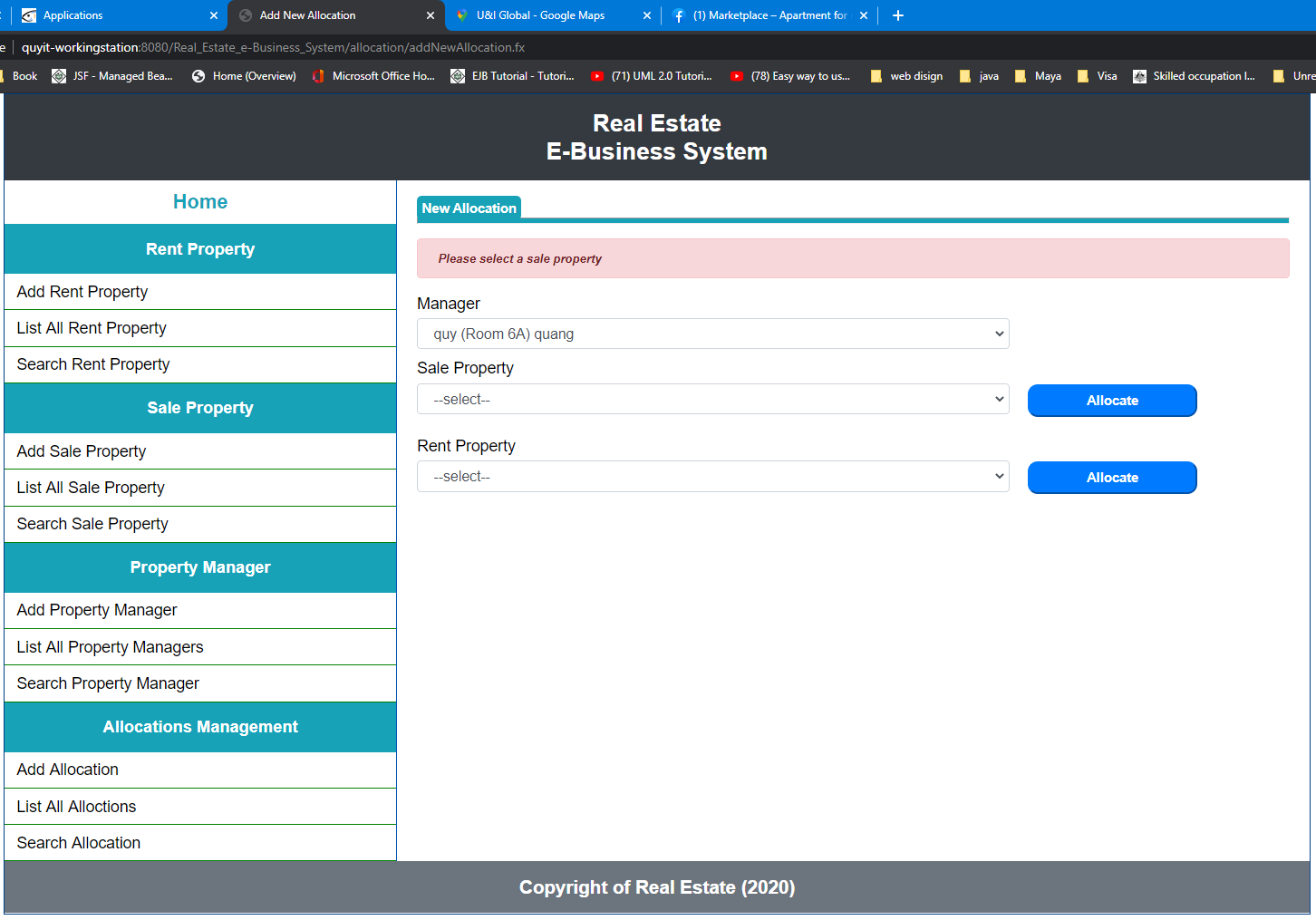


***Save allocation:***

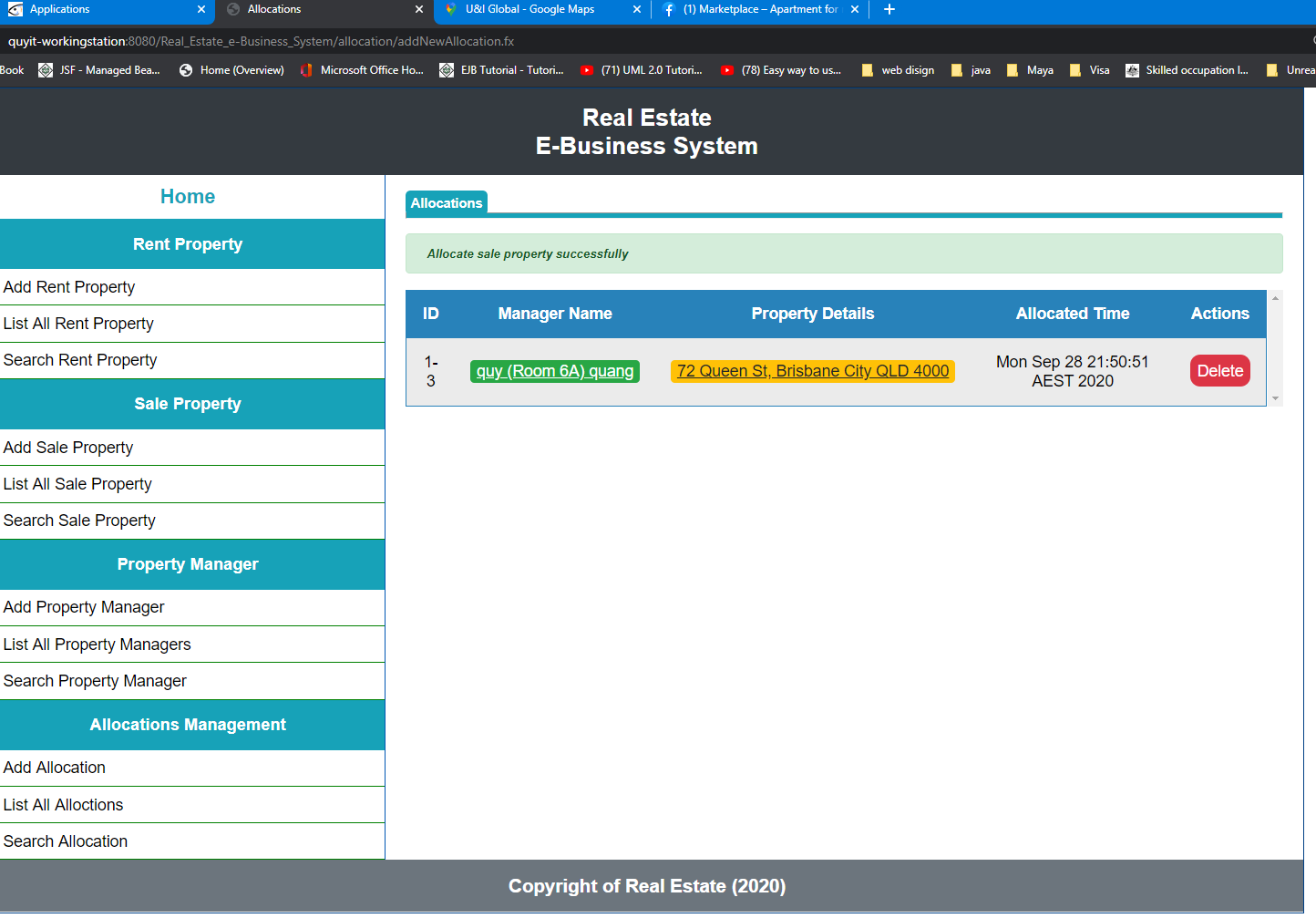
***Not select property manager:***



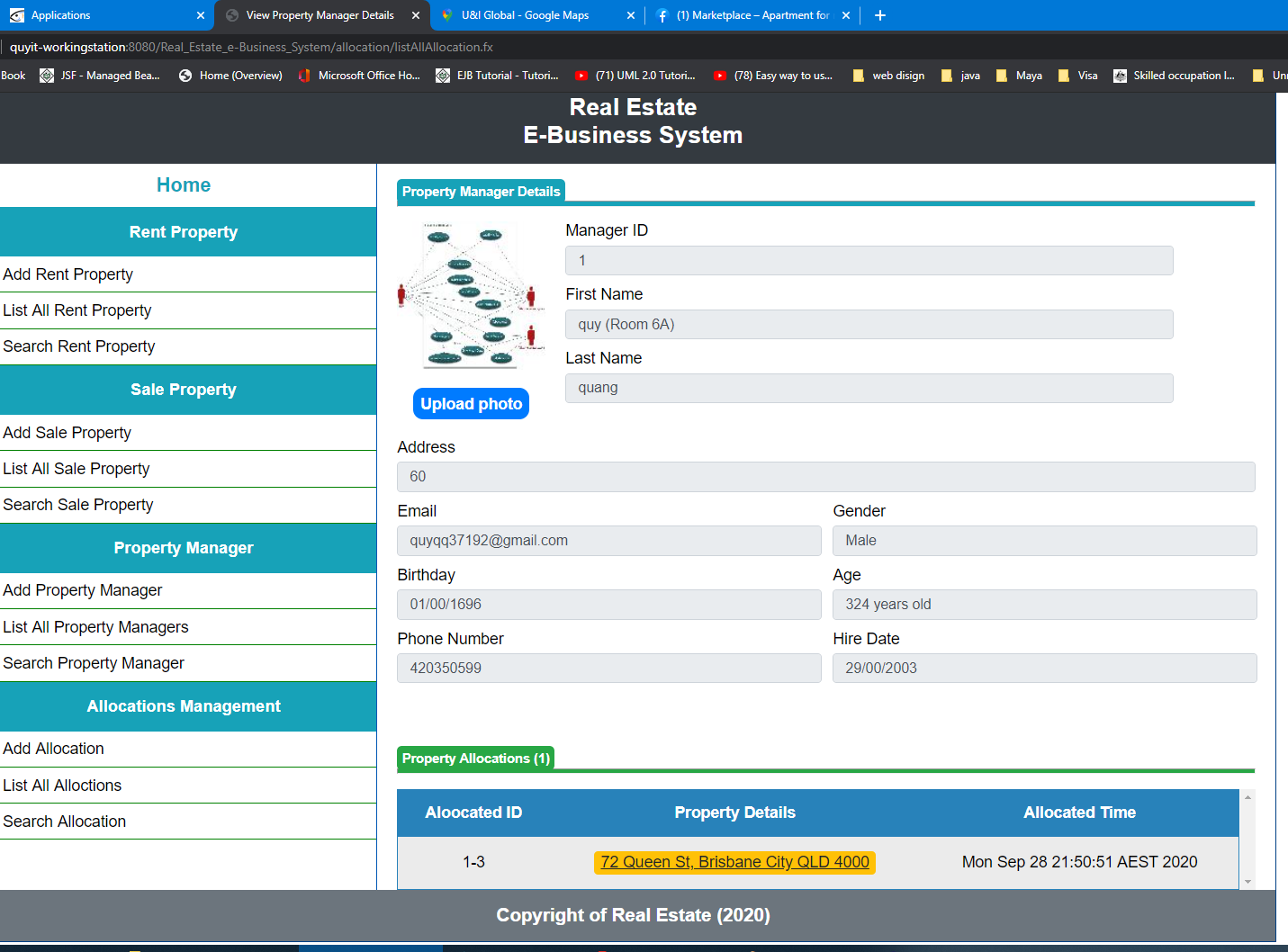
***Not select sale property when allocate sale property:***



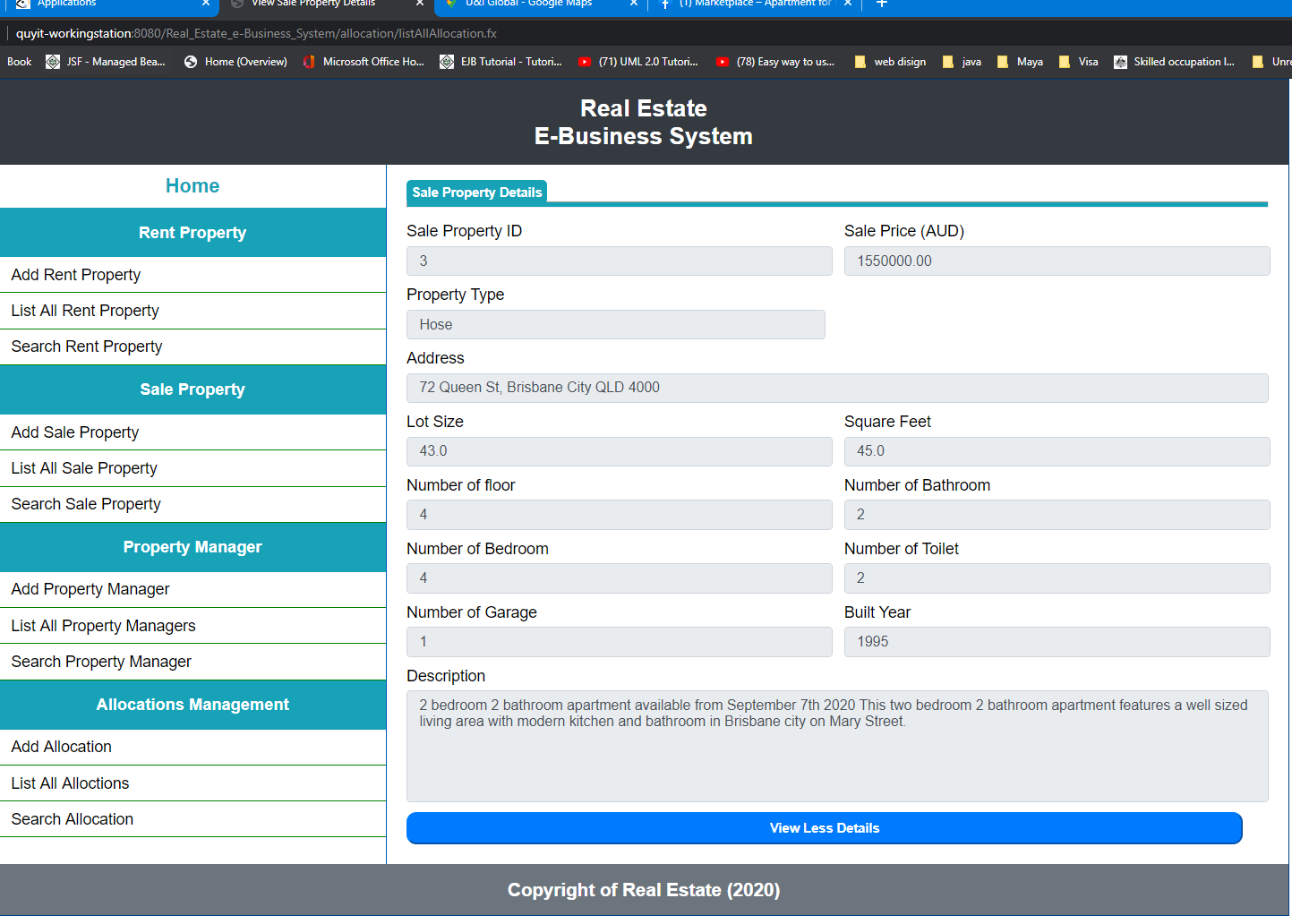
***Successful allocate sale property:***



***Click on manager name:***



***Click on property address:***

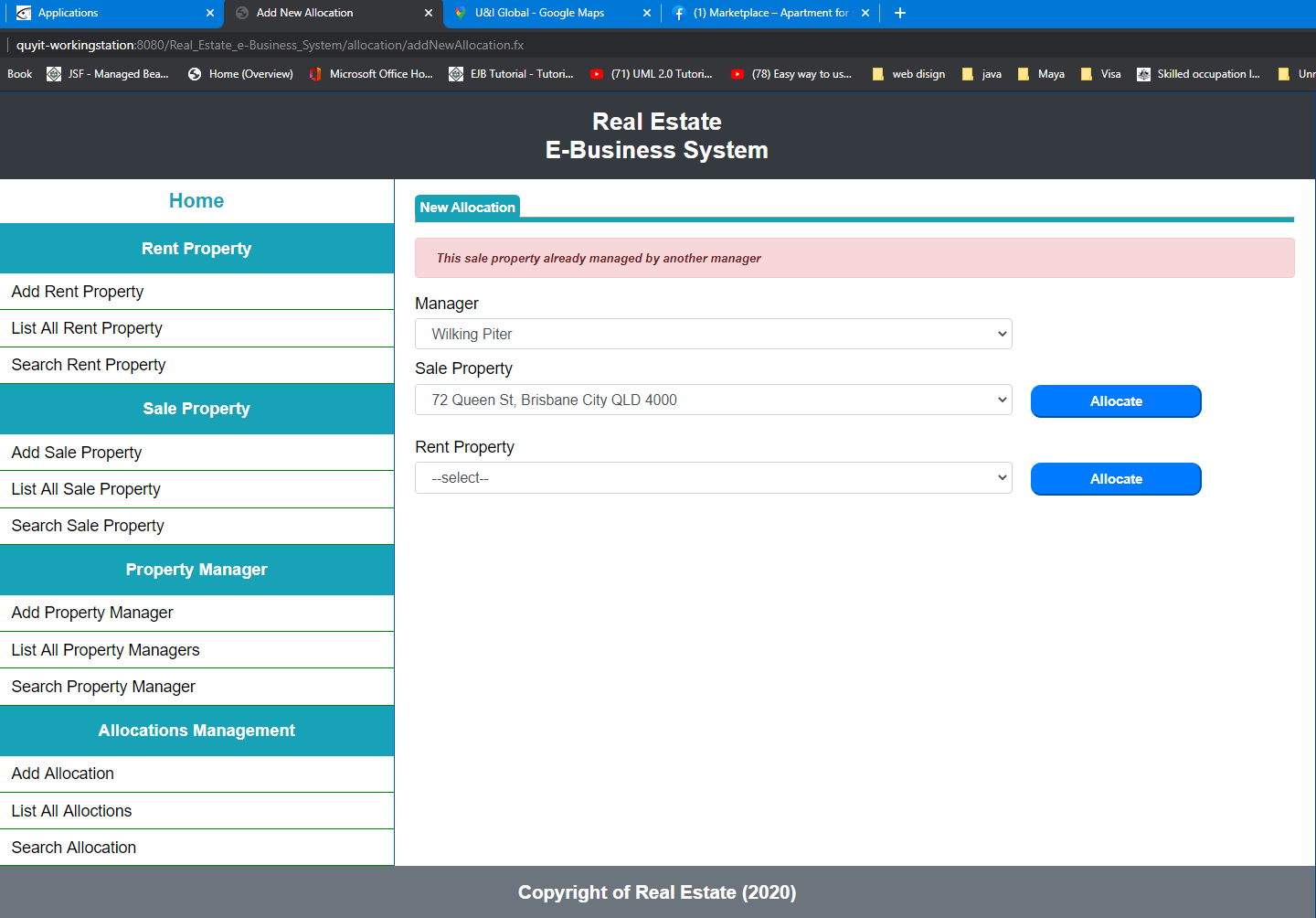


*Click on delete will delete the selected allocation.*

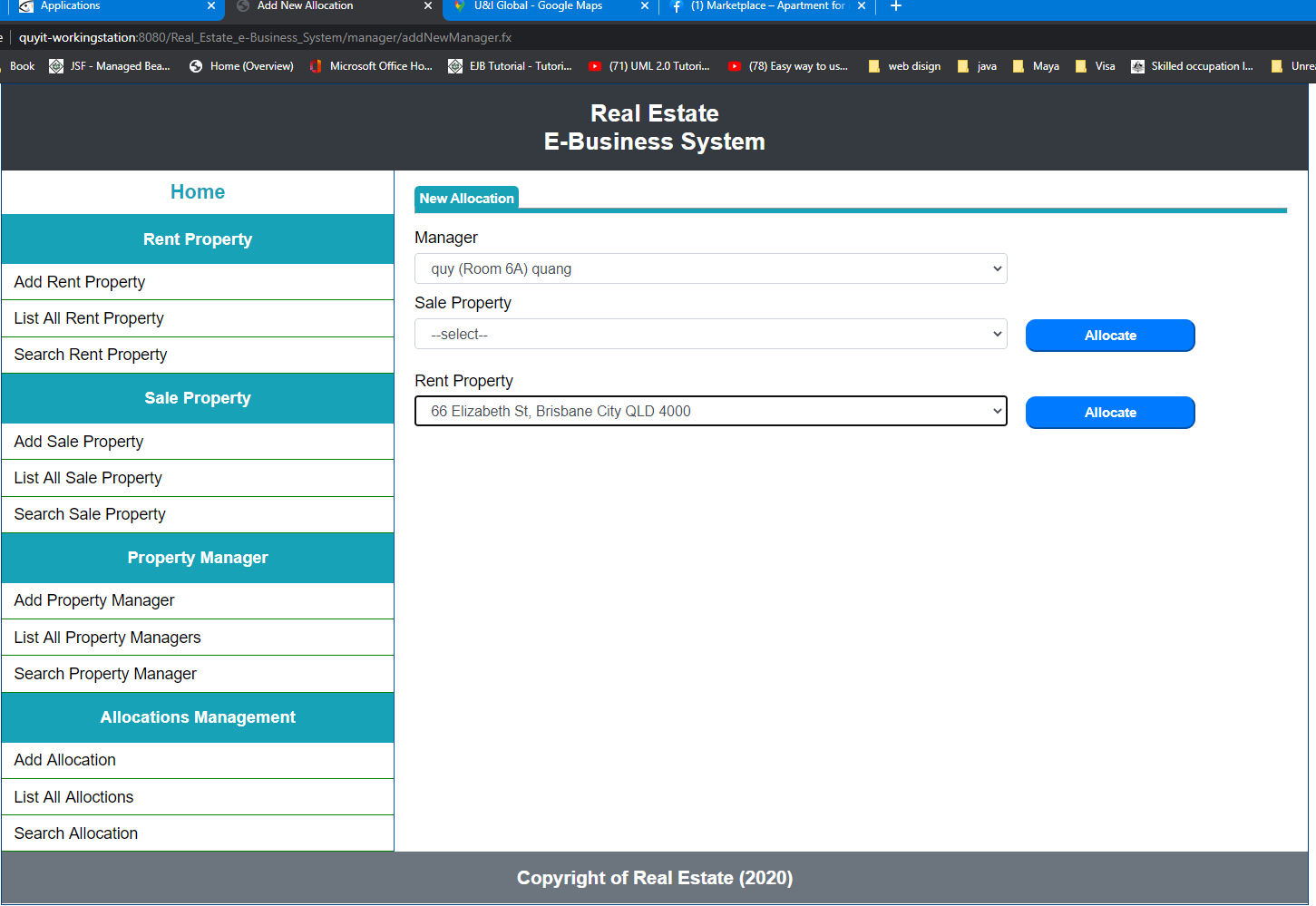
***Allocate sale property which already allocated by the selected property manager:***



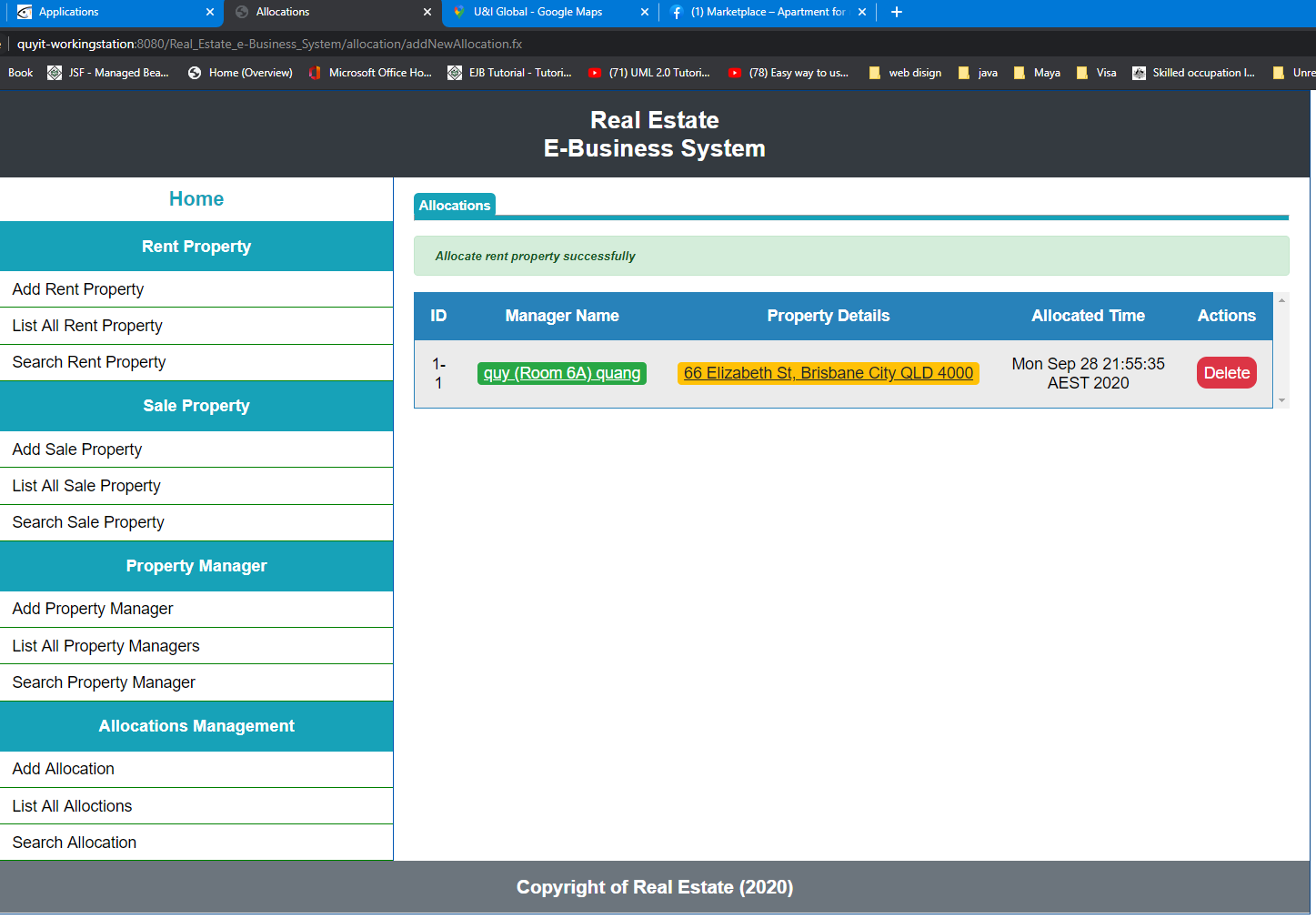
***Allocate sale property which already managed by other manager:***

***`***

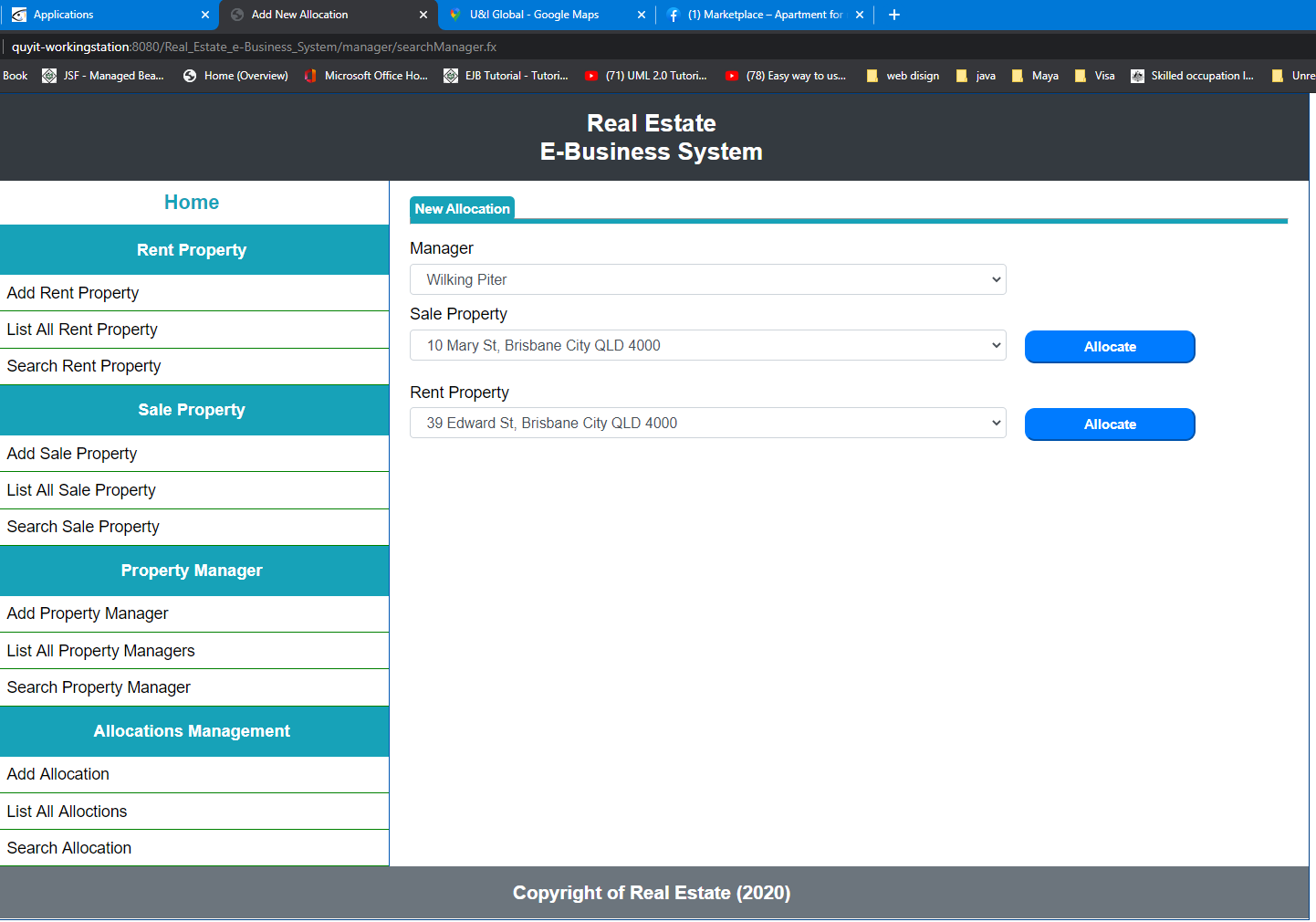
***Allocate second rent property:***



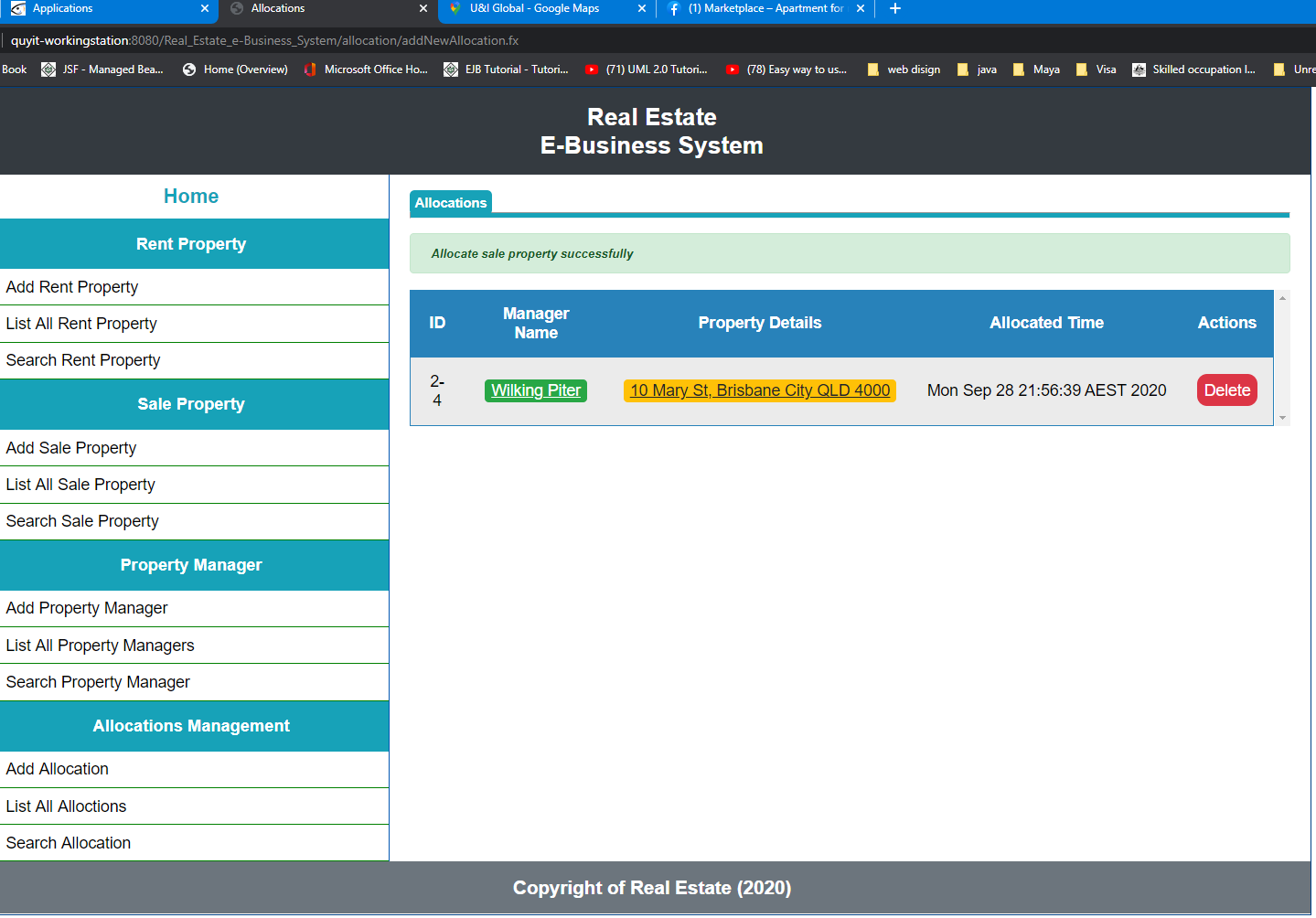
***Result:***

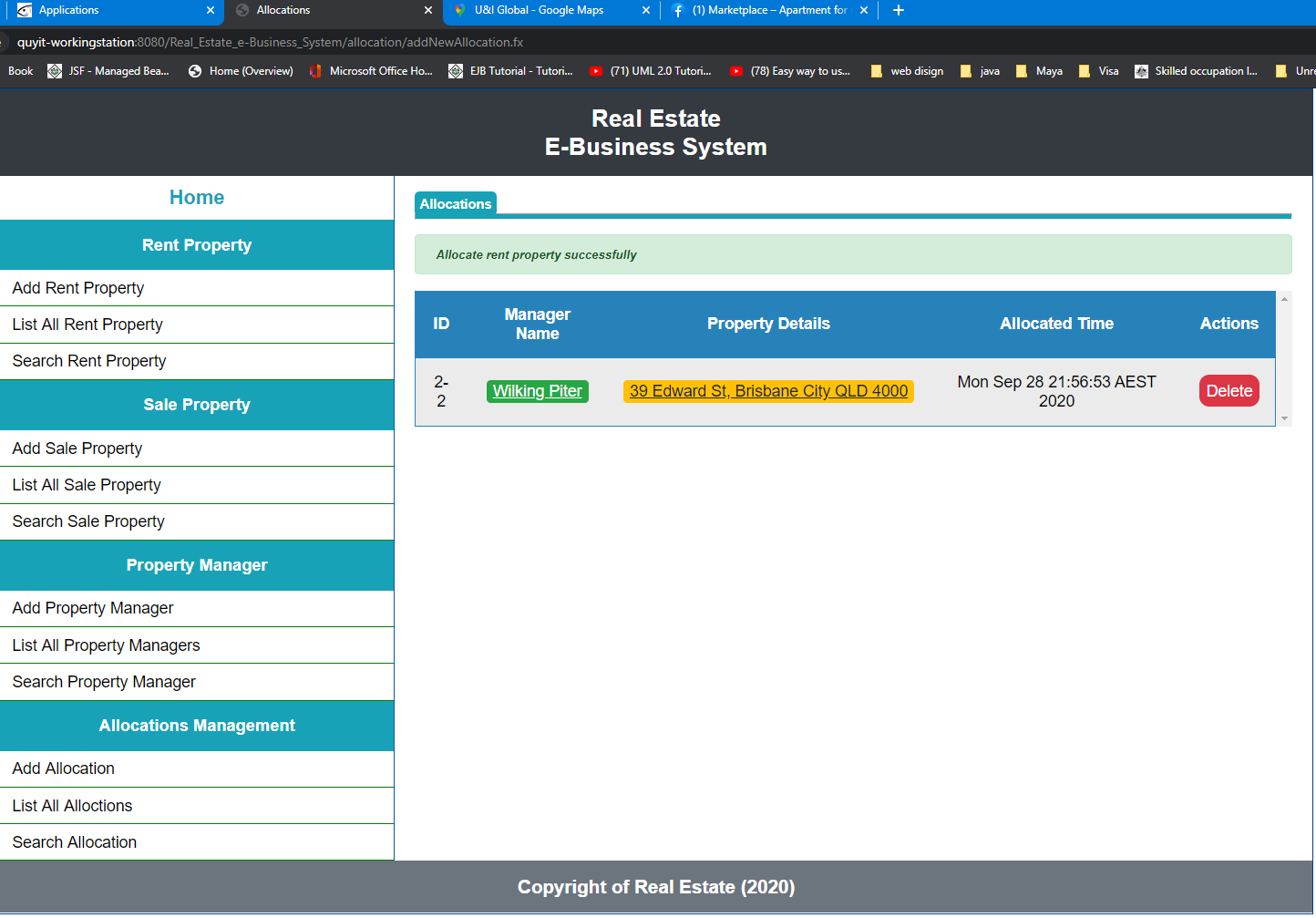


***Allocate for second property manager:***

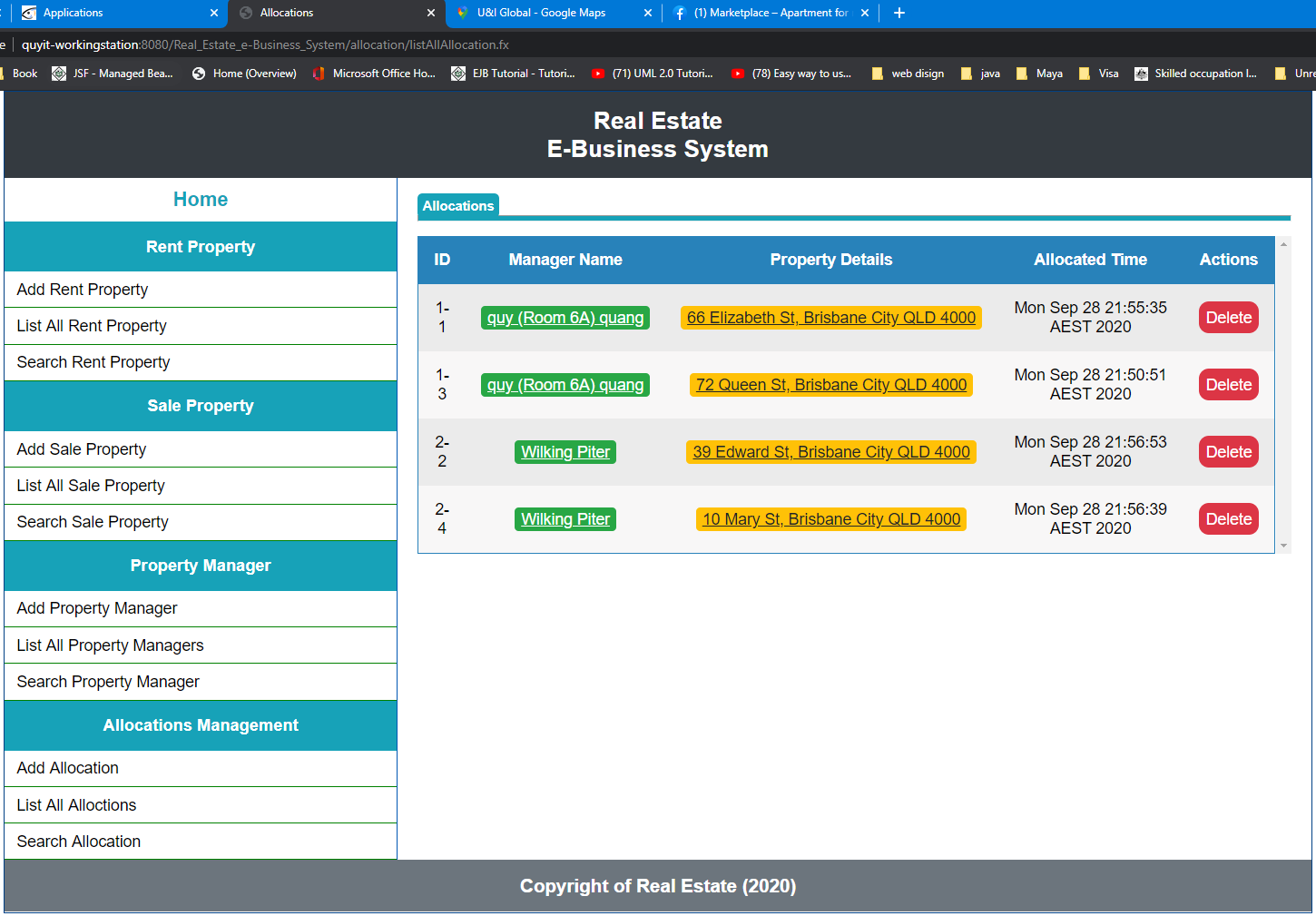


***Result:***



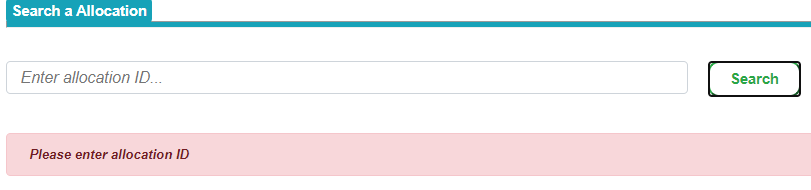


***List all allocations:***

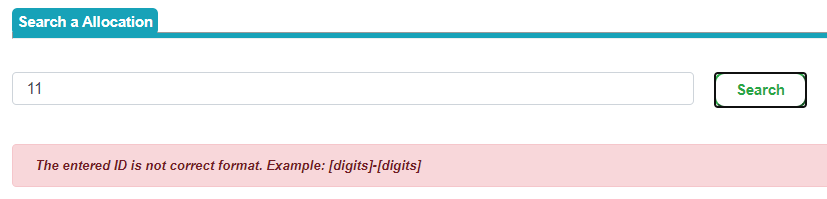


***Search allocation:***

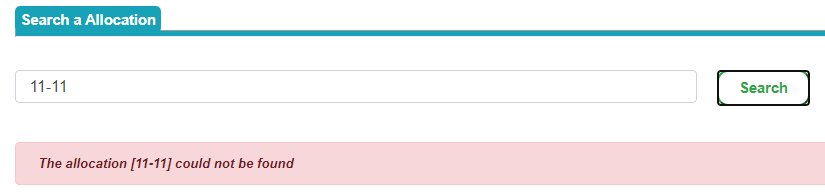
***Empty ID:***



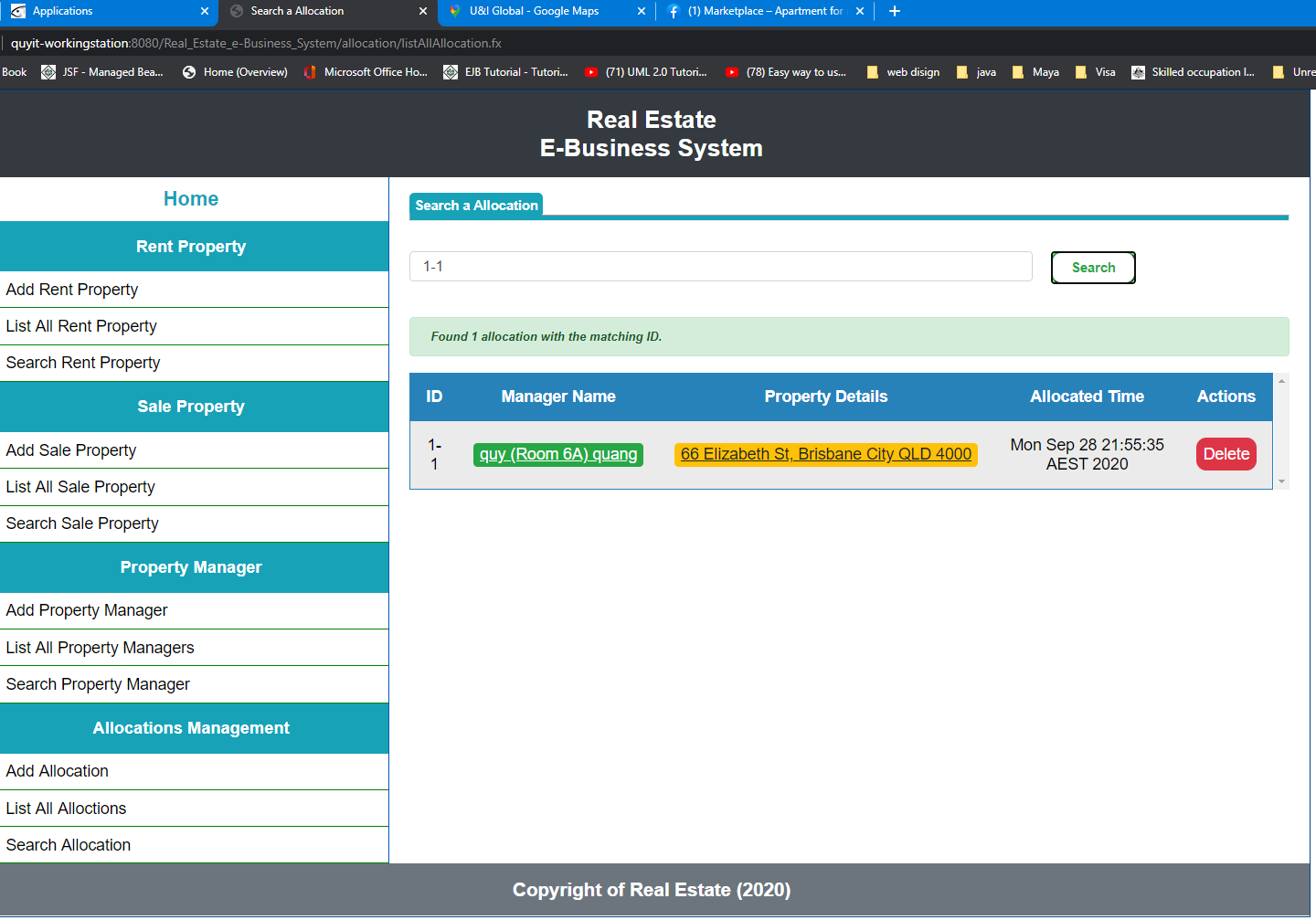
***Incorrect format:***



***Not existing ID:***



***Existing ID:***



***Error page:***

