Developer Tutorial

- **Introduction**

  1- Project Overview:

  Our system called **Carpentry Management** uses cutting-edge technologies such as Java, JavaFX, Dall-E API, and Agile methodologies to provide a range of functionalities that empower carpenters to manage their projects more effectively. You can find also a UML Class Diagram for the project in github when you download the project

  2- Here are some of the key features:

  1️⃣ Visualize Project Requirements: An AI module that generates visual examples rooted in project specifications, making abstract concepts tangible.

  2️⃣ Communication Suite: The system facilitates efficient communication via email functionality with customers, ensuring a seamless flow of information.

  3️⃣ Project Creation & Monitoring: Allows carpenters to initiate, view, and edit project details, maintaining an eagle eye on all facets of their project.

  4️⃣ Inventory Management: Monitors inventory levels, ensuring materials are always stocked for ongoing projects.

  5️⃣ Customer Relationship Management: Enables viewing and editing of customer information, enhancing the relationship management process.

  6️⃣ Suppliers Management: This feature empowers carpenters to meticulously select and manage their suppliers. With an

intuitive interface, users can compare suppliers, evaluate their offerings, and choose the best ones to work with for ordering materials. This not only ensures quality but also aids in cost management, allowing carpenters to maintain an optimal balance between cost and quality.

7️⃣ Settings: The system comes with a customizable settings screen, providing users with the ability to take control of various aspects of the application. Whether it's adjusting notification preferences, modifying display settings, or managing user access, the settings screen allows for a tailored user experience, ensuring that the application aligns perfectly with the carpenter’s workflow and preferences.

8️⃣ Stock: The dedicated stock screen serves as a control center for managing wood information. It provides detailed insights into the types, quantities, and conditions of wood available in stock, enabling carpenters to make informed decisions about material usage. With real-time updates and easy-to-navigate interfaces, managing wood stock becomes a hassle-free task, contributing to the smooth progression of projects.

- **Prerequisites**
  - ➢ Knowledge or skills that are necessary to understand the tutorial:
    1. Object-oriented programming - JAVA.
    2. Eclipse IDE for Java developers.
    3. JavaFX - Scene Builder.
    4. MySQL Workbench 8.0 CE - Database.

  - ➢ System Requirements:
    1. Microsoft Windows Operating System / Apple MacOS.

➢ Dependencies:

1. Access and download the source code by visiting the following link: https://gitlab.com/u_of_haifa_is_dpt/capstone_iris_reinhartz_beregr/Awni-wood-work

2. Click on the link and download the jar files to add them to the project. https://drive.google.com/drive/folders/1anbpK3-5xjBrtTdSAXJYhy7y2mxGu6YJ?usp=sharing

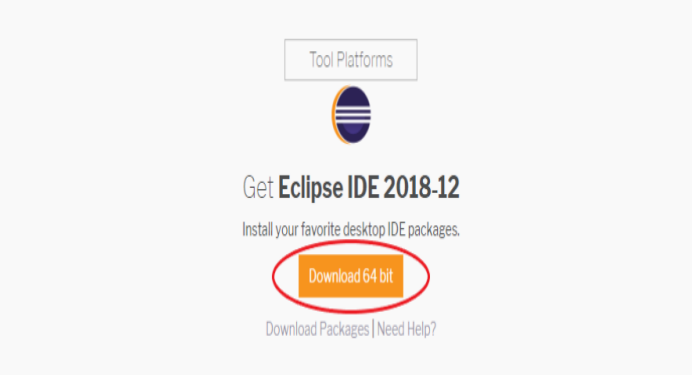
   Or download the latest version of JavaFX from the internet.
   - To download JavaFx JARs click on the link then choose the relevant OS. JavaFX - Gluon

   - Depending on when you read this, the JARs may have been updated. Please check the drive link to obtain the names of the JARs and verify if newer versions are available.

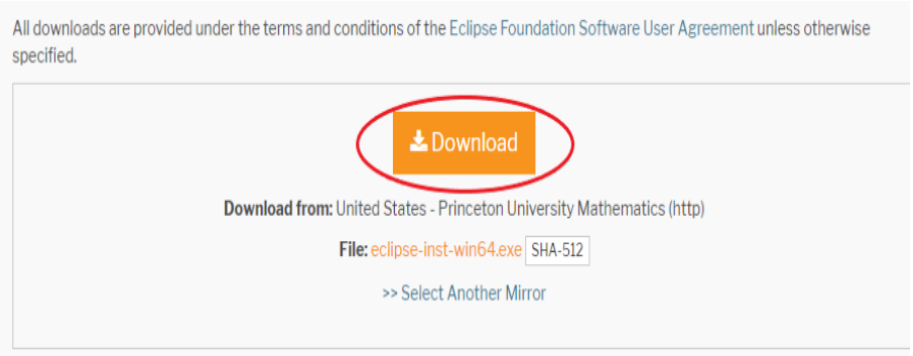
3. Click on the link to download Python. https://www.python.org/downloads/

   You can watch this video for guidance on installing Python on your PC.
   How to install Python 3.10.0 on Windows 11

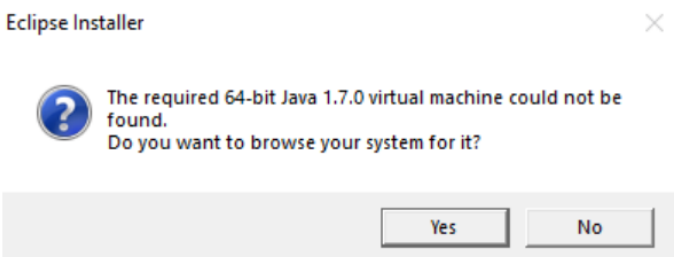
- **Installation & Setup**
  - ➢ Go to the following link: https://www.eclipse.org/downloads/ and click on the marked button which will take you to the download page.



  - ➢ Click on the download button.

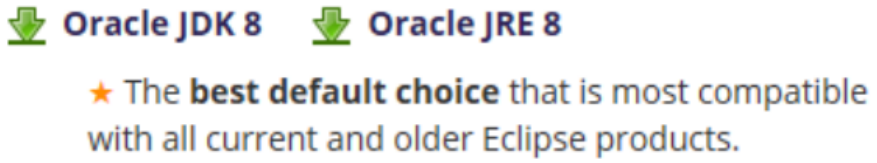


  - ➢ After running the installation file, you may encounter the following error (if the installation process started correctly, proceed to the next step):



To fix this error, you need to install the JDK and the JRE. Go to the following link: https://download.eclipse.org/oomph/jre/index-handler.php

➢ On this page, look for the download for JAVA 8.



➢ Click on the link to download the JDK, accept the terms of use, and download the version suitable for your computer.

Java SE Development Kit 8u201

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ◉ Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	72.98 MB	jdk-8u201-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.92 MB	jdk-8u201-linux-arm64-vfp-hflt.tar.gz
Linux x86	170.98 MB	jdk-8u201-linux-i586.rpm
Linux x86	185.77 MB	jdk-8u201-linux-i586.tar.gz
Linux x64	168.05 MB	jdk-8u201-linux-x64.rpm
Linux x64	182.93 MB	jdk-8u201-linux-x64.tar.gz
Mac OS X x64	245.92 MB	jdk-8u201-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	125.33 MB	jdk-8u201-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	88.31 MB	jdk-8u201-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	133.99 MB	jdk-8u201-solaris-x64.tar.Z
Solaris x64	92.16 MB	jdk-8u201-solaris-x64.tar.gz
Windows x86	197.66 MB	jdk-8u201-windows-i586.exe
Windows x64	207.46 MB	jdk-8u201-windows-x64.exe

➢ Click on the link to download the JRE, accept the terms of use, and download the version suitable for your computer.

Java SE Runtime Environment 8u202

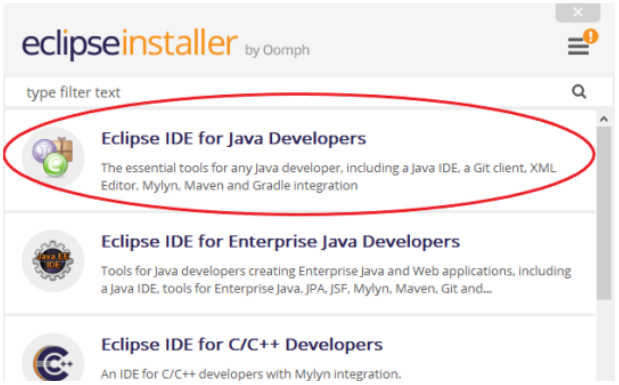
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ◉ Decline License Agreement

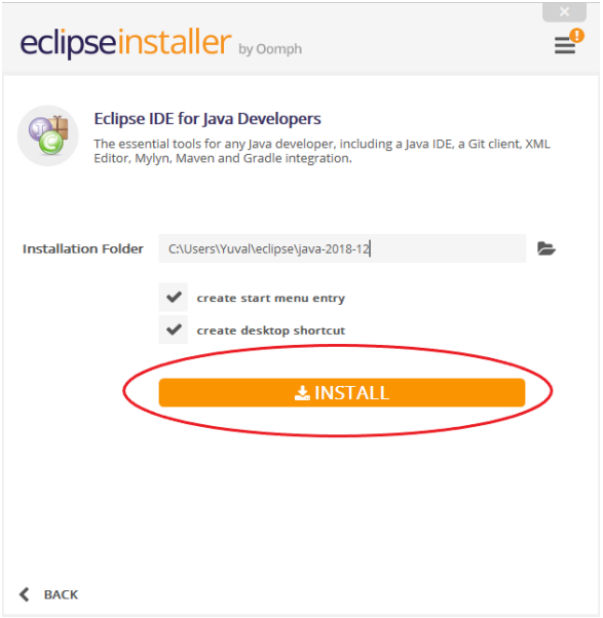
Product / File Description	File Size	Download
Linux x86	70.22 MB	jre-8u202-linux-i586.rpm
Linux x86	85.93 MB	jre-8u202-linux-i586.tar.gz
Linux x64	67.04 MB	jre-8u202-linux-x64.rpm
Linux x64	82.86 MB	jre-8u202-linux-x64.tar.gz
Mac OS X x64	77.79 MB	jre-8u202-macosx-x64.dmg
Mac OS X x64	69.37 MB	jre-8u202-macosx-x64.tar.gz
Solaris SPARC 64-bit	46.07 MB	jre-8u202-solaris-sparcv9.tar.gz
Solaris x64	43.36 MB	jre-8u202-solaris-x64.tar.gz
Windows x86 Online	1.83 MB	jre-8u202-windows-i586-iftw.exe
Windows x86 Offline	65.73 MB	jre-8u202-windows-i586.exe
Windows x86	68.4 MB	jre-8u202-windows-i586.tar.gz
Windows x64	73.7 MB	jre-8u202-windows-x64.exe
Windows x64	73.25 MB	jre-8u202-windows-x64.tar.gz

➢ After downloading the files, run each of them separately and follow the installation instructions.

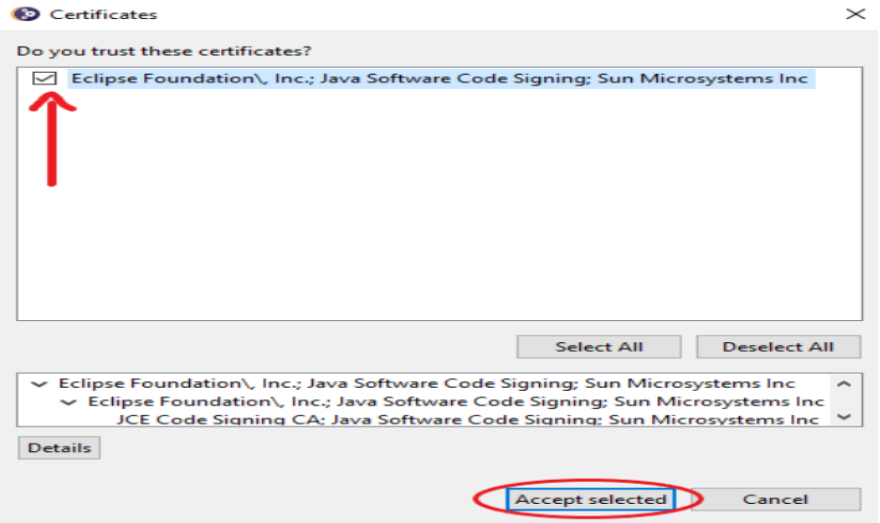
➢ After running the installation, a window will open with several options. Choose the marked option (and accept the terms of use):



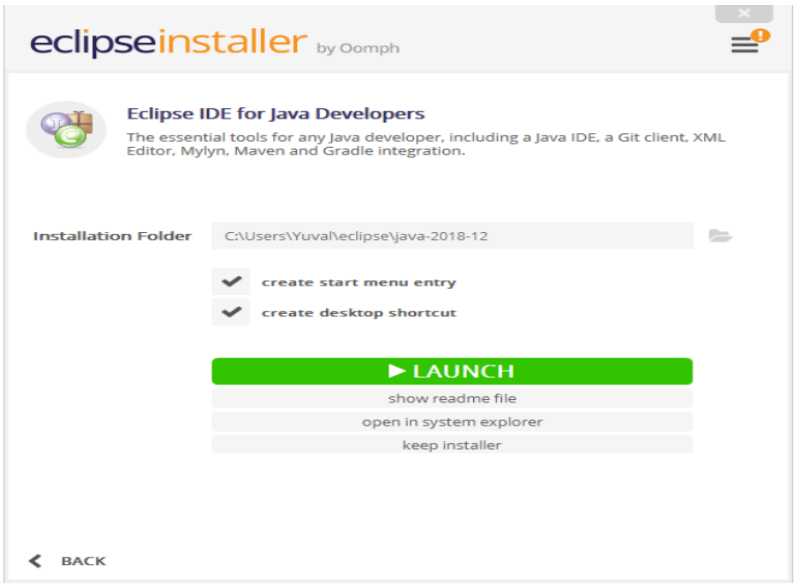
➢ Click on install.



➢ During the installation, you will be required to approve several things:

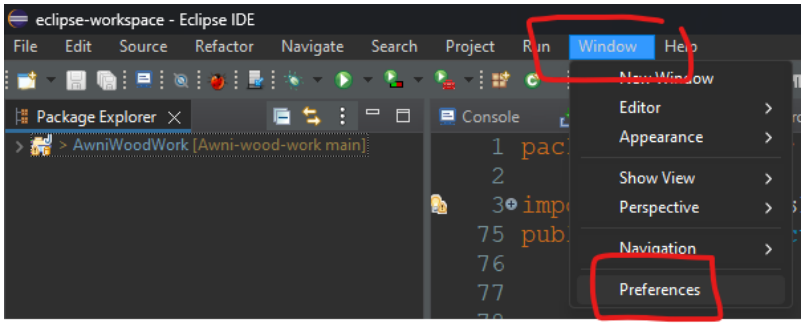


➢ Click on Lunch.

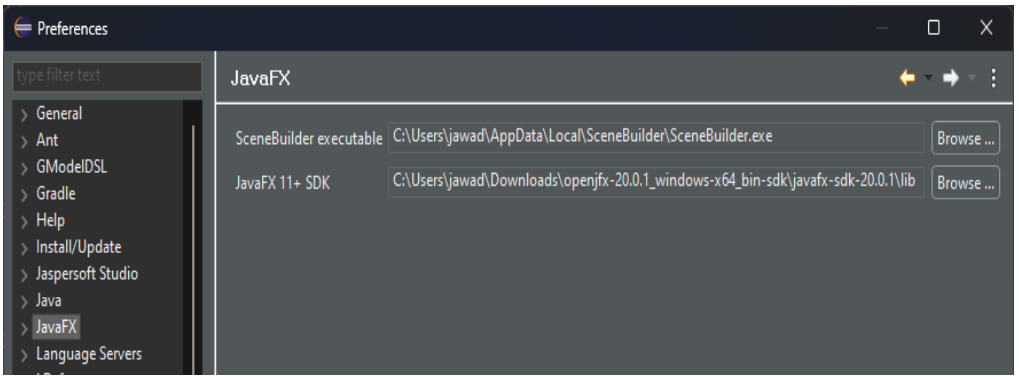


➢ Click on the link to download the scene builder: [Scene Builder - Gluon]

➢ Click on the window then on Preferences.



➢ Click on JavaFx and change the first Path to scene builder path on your PC.
And the second path to JavaFx JARs.



➢ To add the JARs to the project:

- Right-click on the project.
- Click on **Build Path**.
- Click on **Configure Build Path**.
- On the lift side choose **Java Build Path**.
- On the right side choose **Libraries.**
- Click on Modulepath and click on **Add External JARs.** Add JavaFx JARs.
- Click on Classpath and click on **Add External JARs,** add these JARs:
    1. activation.jar
    2. bouncy-castle-connector-8.0.1.jar
    3. commons-8.0.1.jar
    4. commons-io-2.11.0.jar
    5. io-8.0.1.jar
    6. itextpdf-5.5.9.jar
    7. jackson-core-2.15.2.jar
    8. jackson-databind-2.15.2.jar
    9. jackson-datatype-jsr310-2.15.2.jar
    10. javax.mail.jar

11. kernel-8.0.1.jar
12. layout-7.0.1.jar
13. layout-8.0.1.jar
14. mysql-connector-j-8.0.1.jar
15. slf4j-api-1.7.36.jar

Then click apply.

➢ To use MySQL Workbench 8.0 CE as your database, please click on the provided link and follow the steps outlined there.
How to install MySQL 8.0.33 Server and Workb…

After installing MySQL Workbench, in every function change this line
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/DATABASE NAME","User Name","Password");

➢ On Eclipse click on help then on Eclipse MarketPlace, and install:
1- JavaFx. “e(fx)clipse”
2- Python. “PyDev - Python IDE for Eclipse”

- **Core Concepts**
  ➢ Architecture:
  Model-View-Controller (MVC)
  Our project adheres to the Model-View-Controller (MVC) architectural pattern, which facilitates the organization of code in a way that separates concerns, thus

making the project more modular, easier to maintain, and scalable.

- Model: This component is responsible for the data-related logic of the application. It communicates with the database and updates the View whenever the data changes. In our project, the Model contains all the business rules, logic, and functions, ensuring that the data is correctly processed and stored.

- View: The View represents the user interface elements and presentation of the application. It displays data from the Model to the user and sends user commands to the Controller. In our project, the View is responsible for rendering the user interface and displaying the data provided by the Model in a user-friendly manner.

- Controller: Acting as an intermediary between the Model and the View, the Controller receives user inputs from the View, processes them (with possible updates to the Model), and returns the output display to the View. In our project, the Controller handles user interaction, and input validation, and communicates between the Model and the View.

Interaction

In our MVC architecture, the components interact in the following way:

- User Interaction: The user interacts with the View, triggering events related to user input.
- Controller Processing: The Controller receives these events and processes them, possibly modifying the Model as a result.
- Model Update: If there are any changes to the Model, it updates the View accordingly.
- View Update: The View receives updates from the Model and re-renders the user interface to reflect any changes in the underlying data.

The model package includes

1. Axles class - The Axles class is designed to handle an axle object.

2. BackgroundColorEvent class - The Axles class is designed to handle the background of a PDF file.
3. ProjectsToShow class - The ProjectsToShow class is designed to handle a ProjectsToShow object.

4. Customer class - The Customer class is designed to handle a Customer object.

5. Email class - The Email class is designed to handle an Email object.

6. GlobalProjectID class - The GlobalProjectID class is designed to handle a GlobalProjectID object to send an id variable from one screen to another.

7. Hands class - The Hands class is designed to handle a Hands object.

8. OrderedMaterials class - The OrderedMaterials class is designed to handle an OrderedMaterials object.

9. Project class - The Project class is designed to handle a Project object.

10. ProjectDetailsToShow class - The ProjectDetailsToShow class is designed to handle a ProjectDetailsToShow object to show on a specific screen.

11. ProjectDetailsToShowNonStatic class - The ProjectDetailsToShowNonStatic class is designed to handle a ProjectDetailsToShowNonStatic object to show on a specific screen.

12. ProjectItems class - The ProjectItems class is designed to handle a ProjectItems object.

13. Section class - The Section class is designed to handle a Section object.

The view package includes

1. AI_Auto.fxml
2. AI.fxml
3. Email.fxml
4. MainScreen.fxml
5. Menu.fxml
6. NewProject.fxml
7. ProjectCost.fxml
8. ProjectDetails.fxml
9. ProjectImage.fxml
10. ProjectItems.fxml
11. ProjectNotes.fxml
12. Projects.fxml
13. Settings.fxml
14. Stock.fxml
15. Supplier.fxml

The control package includes

1. **AiAutoController:**

   MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

   UpdateProjectImage is a mouse event function that sets the image that the user chooses for the project we are working on.

2. **AiController:**
   MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

   UpdateProjectImage is a mouse event function that sets the image that the user chooses for the project we are working on.

   GenerateImages is a mouse event function that generates images that the user chooses for the project we are working on.

3. **EmailController:**
   MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

4. **mainScreenConroller:**
   LogIn action event that checks if the password that the user entered is correct or not and moves him to the next screen if the password is correct.

5. **menuController:**
   MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

6. **newProjectController:**
   MoveTo is a mouse event function that switches

between scenes when the user clicks on specific buttons.

showCustomer is an Action Event that shows the customer details after choosing an existing customer from the combo box.

update_CustomerDetails is an Action Event that updates the customer details in case the user wants to change existing customer data.

“Case” **projectDetails** checks if the data that the user inserts is valid and if so creates a new customer with the details that the user entered and moves him to the next step to add the project section and the items.

7. **NotesController:**
   UpdateNotes action Event that adds the user's note to the project that he is creating.

8. **ProjectDetailsToShowController**:
   MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

   postInitializedSetup is a function that delivers the data from the previous scene to the current one and gets the selected project details to show it in the table view.

   SubmitProjectDetails action event that updates the data of a specific project after

choosing it from the table view and updating its data.

NewSectionEvent is a function that able or disables the suitable text fields when the user wants to add a new section to the project.

DeleteProject is a function that deletes the chosen project from the database.

DeleteSection is a function that deletes the chosen section from the database.

DeleteItem is a function that deletes the chosen project from the database.

ShowProjectDetails is a function that shows the project details in the table view.

CheckIfDelete is a function that is able and disable the suitable text fields when the user wants to delete/edit a section/project/item to the project.

ShowProjectImage is a function that opens a new scene and shows the selected image for a project.

GetProjectReports is a function that makes a PDF report for the user about a specific project.

9. **ProjectImageController:**

   setData is a function that receives data from the previous scene to another scene.

10. **ProjectItemsController:**

    MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

    AddAnotherItem is a mouse Event that adds a new item to the project.

    AddSection: a mouse Event that adds a new section to the project and calculates the cost and the suggested cost of this project.

    Finish a mouse event that inserts all the data for the project and moves to the next scene.

11. **ProjectsToShowController:**

    MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

    Search / SearchByLetter / SearchEmailByLetter functions enable the user to search for the projects that he needs by letter depending on the field that he chose.

    removeFilter is a function that removes all the filters that the user searched by and shows all the projects in the table view.

DeleteProject is a function that deletes a specific project and all its sections and items.

ShowProjectDetails is a function that helps the user to see all the details of a specific project after choosing it in the table view.

update_Status is a function that updates the status of a project in the database after updating it by the user.

GetProjectReportByMonth is a function that makes a PDF report about all the projects in a month that the user chooses and shows the revenue in that month.

Open_notes is a function that opens a new scene to show the user the notes of a specific project.

12. **SettingsController:**

MoveTo is a mouse event function that switches between scenes when the user clicks on specific buttons.

AddORDeleteButton is a function that adds/deletes data to the database depending on what the user wants to add(woodType/ axles/hands/supplier).

13. **StockController:**

MoveTo is a mouse event function that switches between scenes when the user clicks on

specific buttons.

UpdateWoodQuantity is a function that checks if an order exists that the user wants to update in the database if yes it updates the quantity of the wood in the database.

order_Wood is a function that makes a new order with the user's preferred quantity.
ShowStock is a function that shows the stock data in the table view.

outOfStock is a function that checks if there is enough wood in the stock when the user makes a new order.

14. **SupplierController:**

setData is a function that receives data from the previous scene to another scene.

GetSuppliersEmails is a function that gets the supplier's Emails from the database to help the user make automatic orders when there are not enough materials in stock.

15. **CarpentryLogic:**

addCustomer to add customer object to the database.

getCustomer to get customer objects from the database.

updateCustomer to update customer information.

updateCustomerEmailInProject to update the customer email in a specific project.

addAxles to add an axle object to the database.

getAxles to get axle objects from the database.

DeleteAxle to delete Axle objects from the database.

updateAxleQuantity to update axle quantity.

updateAxleDegree to update axle degree.

addHands to add a hand object to the database.

getHands to get hand objects from the database.

DeleteHands to delete Hand objects from the database.

updateHandsQuantity to update hands quantity.

updateModelOfHands to update the model of hands.

addSupplier to add a Supplier object to the database.

getSupplier to get Supplier objects from the database.

DeleteSupplier to delete Supplier objects from the database.

addWoodType to add a WoodType object to the database.

getWoodType to get WoodType objects from the database.

DeleteWoodType to delete WoodType objects from the database.

updateWoodType to update item wood type.

addProjects to add a Project object to the database.

getProjects to get Project objects from the database.

DeleteProject to delete a project from the database.

updateProjectStatus to update project status.

updateProjectNotes to update project notes.

GetImage to get the image for a specific project.

DeleteSectionImage1 to delete the image of the kitchen section in a specific project.

DeleteSectionImage2 to delete the image of the Room section in a specific project.

DeleteSectionImage3 to delete the image of the Livingroom section in a specific project.

DeleteSectionImage4 to delete the image of the Bathroom section in a specific project.

addProjectItems to add a ProjectItem object to the database.

getProjectItems to get ProjectItem objects from the database.

DeleteProjectItems to delete a project item from the database.

updateItemColor to update item color.

updateItemHeight to update item height.

updateItemWidth to update item width.

updateItemName to update item name.

updateItemQuantity to update item quantity.

updateItemSection to update the item section.

updateItemSectionID to update the item sectionID.

addSections to add a Section object to the database.

getSections to get Section objects from the database.

DeleteSection to delete a section from the database.

updateSectionName to update section name.

addStock to add a stock object to the database.

getStocks to get stock objects from the database.

addOrderedMaterials to add an OrderedMaterial object to the database.

getOrderedMaterials to get OrderedMaterial objects from the database.

DeleteOrderedMaterials to delete OrderedMaterial objects from the database.

updateStockQuantity to update the stock quantity in the database.

updateProjectCost to update project cost.

updateProjectPrice to update project price.

addProjectImage to add an image to a specific project.

16. **<u>ProjectCostController:</u>**
    SetProjectCost to set project cost after adding/deleting a section or item.

- **Troubleshooting**
  - ➢ Common Issues:
    1. The video in the background of the MainScreenController might not be played due to an unidentified loading bug.

    2. The email loading scene might take a few seconds to load because of the threads running in the background.

    3. The AI code may take a few seconds to finish generating, depending on the PC specs and internet speed.

- **API Documentation**
  - ➢ API Overview:

    DALL·E, a brainchild of OpenAI, represents a significant leap in the AI-driven image generation domain. Building upon the architecture of GPT-3, DALL·E is not just another text model. Instead, it is uniquely designed to generate vivid and often surprisingly accurate images from textual descriptions.

    At its core, DALL·E's strength lies in its ability to interpret and artistically render textual prompts into visual art. From simple requests like "a two-headed flamingo" to more abstract concepts like "a futuristic cityscape at sunset," DALL·E can create a wide range of images that often surpass the expectations of its user.

    For developers, the DALL·E API provides a seamless avenue to harness this capability. Whether you're in content creation, game design, digital art, or any field that can benefit from on-the-fly image generation, the API acts as a bridge between human creativity and AI innovation. Integration is streamlined, with OpenAI providing comprehensive documentation to ensure that even those new to the space can get started with ease.

    In conclusion, the DALL·E API is not just a tool—it's a canvas for the next generation of digital artists and developers. As you delve into its capabilities, remember that while the

technology is cutting-edge, the real magic happens when human imagination meets AI potential.

➢ API Endpoints/Methods:
Click on this link:
https://platform.openai.com/account/api-keys

After creating an account, click on the top right side where it says "Personal" to create an API key. Then, click on "View API Keys" and create a new secret key. Copy this key and paste it into line 49 in a package named "AI" within the demo.py file.

On line 55, you can set the path for the generated images.
On line 57, you can set the number of images (num_image) to generate in a single run.
Also on line 57, you can choose from three image sizes to generate by setting the values 0, 1, or 2 in SIZES[X]. Here, X represents the size of the image (as referred to in line 50).

Please check the pricing on the OpenAI site.

We implemented this AI feature in the AiAutoController. This code is executed when the carpenter creates a new project using the new sections and item specifications.
Additionally, we employed it in another scene, AiController, where the code is run using a text field to generate the images.

- **Resources & References**
  - ➢ Related Reading:
    Discover more about our project by clicking the link below!
    CarpentryManagment.pptx

  - ➢ Community & Support:
    For community discussions, support, and insights, reach out directly to our team via email at jawad.makhoul9@gmail.com
    or hasan.masalha98@gmail.com