CS495 Software Project Proposal: ClearView CRM and Window Design Tool

Jack Formato and Chris Plowman

2025-04-03

1   Client Information

By sharing this client information and the rest of this document, you are stating that this client has provided this project as something they want (not something you created and asked if they wanted), and that they are interested in having you complete this project for your capstone.

1. Client name: Deirdre Hammaker

2. Client title: CEO of ClearView Doors & Windows

3. Client email address: deirdre@getclearview.com

4. Client employer: ClearView Doors & Windows

5. How you know the client: Friend of Chris'

2   Project Description

2.1   Overview

This project aims at creating an internal web app for ClearView. The app's main purpose is to help sales representatives and installation crews better manage the job process. The app will let the sales rep show customers previews of their house with newly designed windows based on the customer's selections. These selections include a choice of multiple customization options for windows. This will also let the sales rep show the customer an estimated quote based on their selections. From there the job will go through a series of stages in which the customer will receive email and or text message notifications (customer's choice) about their job updates. At any point, the customer can see the status of their job through the app or a link sent with the notifications.

2.2   Key Features

The app's first key feature is the ability for the sales rep to take a picture of the customer's house using their company iPad, to allow for previews of the windows. This will put a border around the window as well as a tab for the sales rep to select individual windows and edit them to the customer's wants. During this mock up process, the pictures will preview the modified windows and calculate the estimated cost for each window to be used in an estimated quote. Once the sales rep's job is done and the job is accepted, the installation crew will be able to see the job details.

The app will have a status bar for both the customer and the office managers to view. The statuses, or stages, are listed below:

1. Created

2. Viewed

3. Accepted

4. Final Measurements

5. Final Review

6. Order Placed

7. In Production

8. Order Received

9. Installation in Progress

10. Complete

Under each stage will be either the date started, if it is in progress, or the date completed. The customer will be able to cancel their order with no penalties for 72 hours following the acceptance of the project. This will be the limit of the customers view. The office managers will be able to view details of the project surrounding the status: ClearView's cost, sale price and manufacturing details.

The project will have a log-in feature with several access levels. The highest is Admin. Admin accounts will be for office managers. Admins have access to all quotes and projects regardless of who the sales rep was. They will be able to view and change the status of a project. The next level will be Sales Rep. Sales reps will be able to view the status of their active jobs. They will only be able to edit their jobs in the initial stages. These are Created, Viewed, and Accepted. The last company access level will be the installation crew. The installation crew does not have access to the status bar, and is instead told by the office manager what jobs are assigned to them. The installation crew can access the details of the job. This includes things like the name of the customer and sales rep, the details of each window, and the images the sales rep took so that they know which window is which. They will then be able to input measurements and final pricing into the app. Customers are limited to viewing the status of their order and can cancel within 72 hours of the project being accepted.

2.3 Why this Project is Interesting

This project is interesting to us for a few reasons. The first reason is that this project is very practical. Many other companies in this field use apps similar to this. Another reason is that this project has some challenging aspects which intrigue both of us, including working with computer imaging and potentially LLMs. The last aspect that gets both of us excited to work on this project is the possibility that this app that we create could bring real value to the company.

2.4 Areas of CS required

This project will require many areas of CS including; database managing, computer/digital graphics & vision, the example that our client showed us uses an LLM which we will explore using.

2.5 Potential Concerns and Questions

Our potential concerns are mostly related to the feasibility of the project. This is the first time that my partner or I have worked on such a large and practical project. Although we are interested in working with an LLM, we are concerned that it might prove to be too difficult. One of the example sites that our client showed us says that it uses an LLM to do its imaging and mock-ups. Additionally, if we don't use an LLM, how will we accomplish the task of putting borders around windows for the sales reps.

3 Comparison to Draft

Fill out this section by answering the following questions in paragraph form (you can remove these instructions as you complete it):

- This project is the same as the one that **Chris** proposed in his draft.

- This project is not new although it is slightly altered.

- The client is not new.

- This project has changed in the usage of the app. Originally, it was designed for a customer to use. Now it's designed for internal staff of ClearView to use. Additionally, the focus is on windows instead of windows and doors.

4 Requirements

4.1 Functional Requirements - (82pts)

ID	Name	Points	As a	Functionality	Related NFRs
S1	Upload Picture	3	Sales rep	I want to upload a picture of a side of a house so that I can start to edit the windows on that side	1 5 6
S2	Select Window	5	Sales rep / Installation Crew	I want to select a specific window on the picture so that I can customize the window or view details about that window	1 5 6
S3	Customize Window	8	Sales rep	I want to customize the window so that I can preview the finished window on the house	1 2 5 6
S4	Edit Border	2	Sales rep	I want to edit the border of a window so that I can adjust the border of the window as needed	5 6
S5	Manual Add	5	Sales rep	I want to manually add a new window to the preview so that I can add a window the software might have missed	5 6
S6	Remove Window	2	Sales rep	I want to remove a window so that the window's border is no longer on the preview	5 6
S7	Save Window	2	Sales rep / Installation Crew	I want to save an individual window so that the details are saved for later	3 6
S8	Save Project	2	Sales rep	I want to save the entire set of windows so that the entire set of windows are saved for the project	3 6
S9	CRUD Person	8	Sales rep / Admin	I want all CRUD functionality for client information so that I have all CRUD functionality for client information	3 6
S10	CRUD Project	8	Sales rep / Admin	I want all CRUD functionality for projects so that I have all CRUD functionality for projects	3 6
S11	CRUD Project Details	8	Sales rep / Admin	I want all CRUD functionality for project details so that I have all CRUD functionality for project details	3 6
S12	CRUD Account	8	Admin	I want all CRUD functionality for employee accounts so that I have all functionality for employee accounts	3 6
S13	Search Projects	2	Admin	I want to search for a specific project so that I can see information related to that project	3 6
S14	Compact View	3	Admin	I want a compact view of some of the project's details so that I can get a succinct view of the project's details	1 5
S15	Detailed View	3	Admin	I want a detailed view of all the project's details so that I can see all the project's details	1 5
S16	Assign Project	3	Admin	I want to assign a project to an Installation crew (member) so that project has a crew assigned to it for completion	3 5 6
S17	View Assigned Projects	2	Installation Crew	I want to view all projects assigned to me / my crew so I can see which projects are assigned to me / my crew	1 3 5
S18	Select Project	2	Installation Crew	I want to select a project that has been assigned to me / my crew so that I can select which project my team is going to work on	1 3 5
S19	Input Final Window Measurements	3	Installation Crew	I want to input the final measurements for the windows in the project so that the project has accurate measurements	1 5 6
S20	View Project Status	3	Customer	I want to view my project's status bar given the link (email / text) so I can track the progress of my project	1 3 4 5 6

4.2 Non-Functional Requirements

ID	The	Shall Have	With this Feature
NFR1	UI	Colors	that match the company's colors (white & blue)
NFR2	Window Designer	Automated	That will generate a rough estimate of the dimensions of the window
NFR3	System	A database	That uses PostgreSQL
NFR4	CRM	Notifications	That tell the customer when their project's status has updated
NFR5	System	Front end logic	Written in HTML, JavaScript, CSS and React
NFR6	System	Back end logic	Written in Python

4.3 Iteration Planning

- Sprint 1 - (11pts)

  - S1: Upload Picture
  - S9: CRUD Person

- Sprint 2 - (21pts)

  - S2: Select Window
  - S10: CRUD Project
  - S11: CRUD Project Detail

- Sprint 3 - (21pts)

  - S3: Customize Window
  - S4: Edit Border
  - S5: Manual Add
  - S6: Remove Window
  - S7: Save Window
  - S8: Save Project
  - S21: CRUD Window (Added Sprint 3)

- Sprint 4 - ~~(17pts)~~ (9pts)

  - ~~S12: CRUD Account~~
  - S13: Search Projects
  - S16: Assign Project
  - S17: View Assigned Projects
  - S18: Select Project

- Sprint 5 - (12pts)

  - S14: Compact View
  - S15: Detailed View
  - S19: Input Final Window Measurements
  - S20: View Project Status

5 Design & UI

5.1 Architecture

Our project will use a 3-layer design. The first layer of our project will be the frontend. This will be programmed in React, JavaScript, and HTML/CSS. It will handle all the client-side functionality. The second layer, the backend, will be programmed in Python and will be the server that receives information from the frontend. It will contain all our classes, handle sensitive data evaluation, and analyze the images it receives. The backend will fetch data from our final layer which is the database. The database will save account information and projects. Here is a simple UML class diagram for our project.
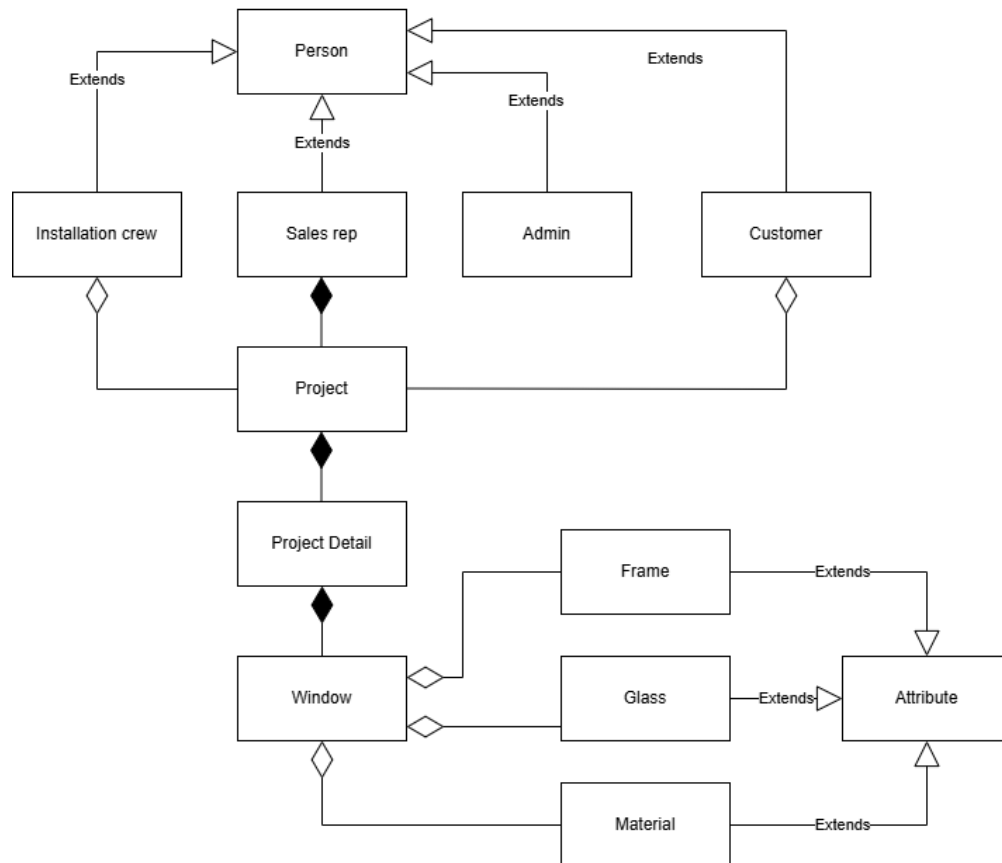
Figure 1: Simple UMLD

5.2 Technology

Here are the languages, libraries and modules we either plan on using or think might be useful.

- Python - Backend

- HTML, JavaScript, CSS, React - Frontend

- Opencv - computer vision, image analyzing

- Matlib

- Django, Flask or Web2py - Web Development

- Psycopg - PostgreSQL

- SMTP Library - text/email

- requests, Pillow - upload image

- pytest

- Jest

5.3   Data

Here is an ER diagram for our database structure.
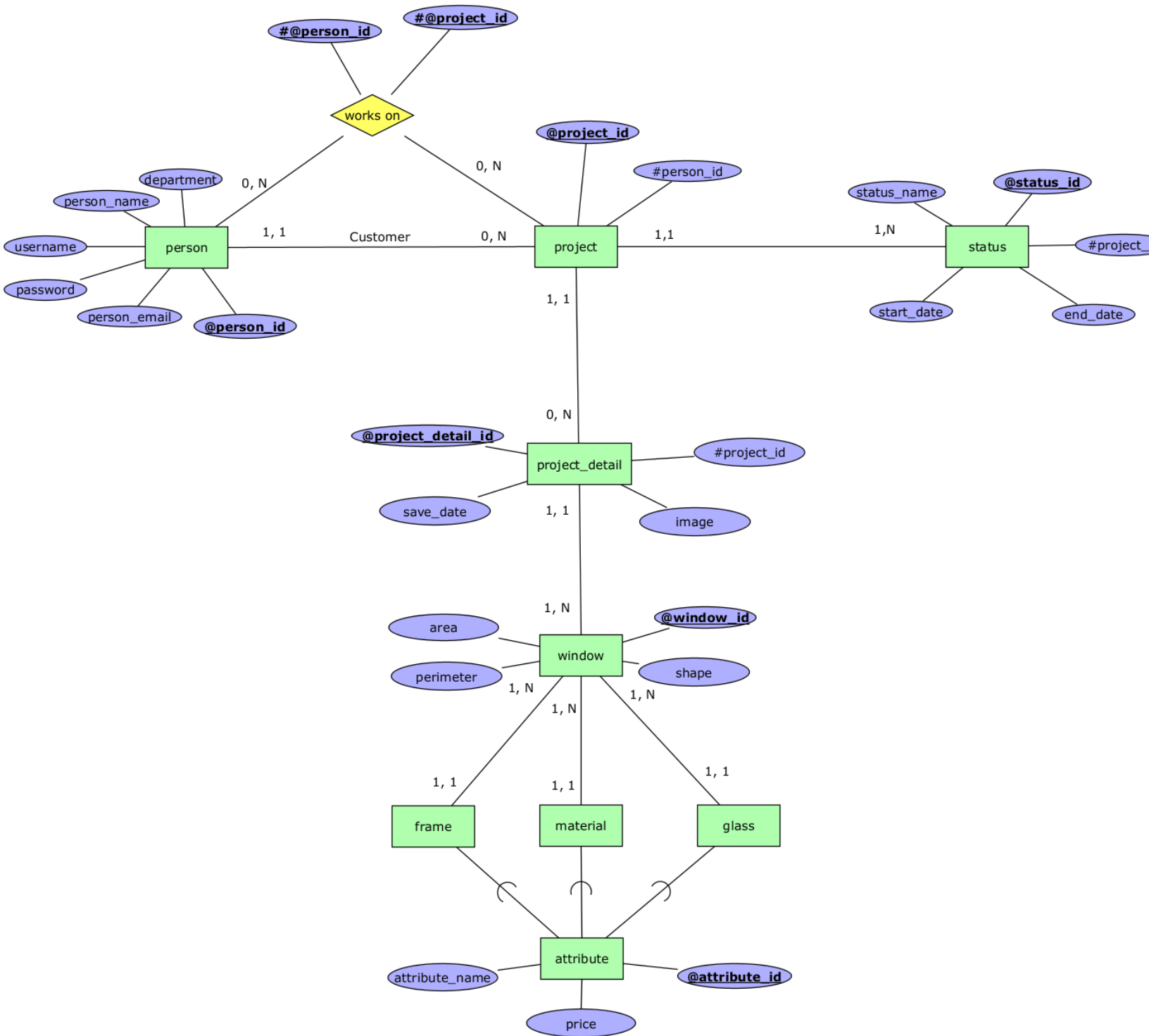
Figure 2: ER Diagram

5.4 Coding Standards

For our project we have chosen to abide by the following standards:

- For Python code, variables and functions will use snake_case while classes will use capitals

- For JavaScript, HTML and CSS code, variables, functions and classes will use camelCase

- Our database will use a lowercase, singular naming convention for the SQL tables and snake_case for both the tables and attributes

- For testing we have agreed on a 70% coverage required for backend code (python) and at least 50% for front end code (HTML, CSS, React)

5.5 UI

Here are some UI mock-ups for a few of the key pages in our project:
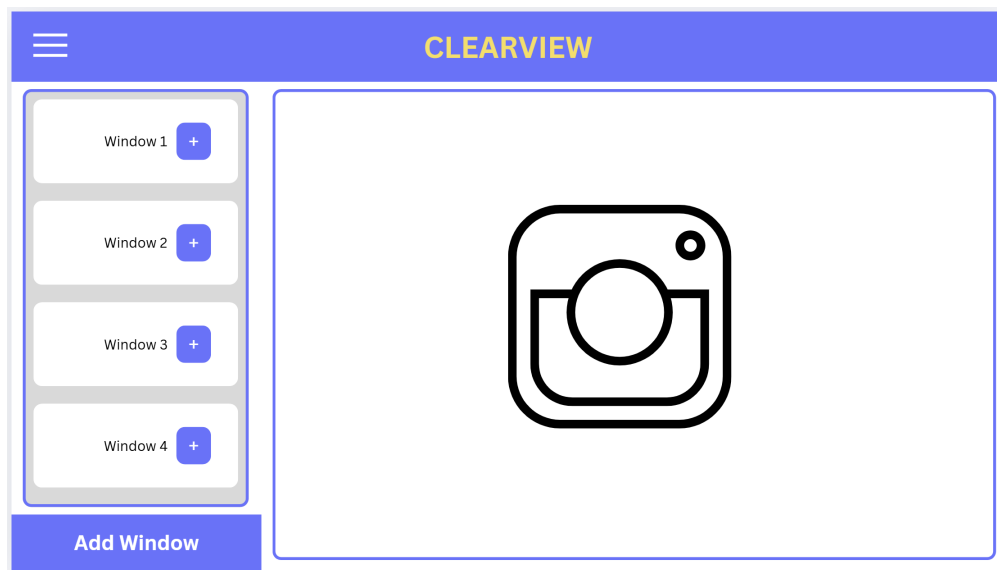
Figure 3: This is the window designer that customizes windows and allows for a preview

Figure 4: This is the list of all the jobs in the system which only an admin can see

Figure 5: This is the detailed view of a particular job which only an admin can see

6 Sprints

6.1 Sprint 1

Table 1: Overview for Sprint 1

ID	Name	Points	Contributor
S1	Upload Picture	3	Chris
S9	CRUD Person	8	Jack
T1	Create Database	2	Jack
T2	Setup Framework	2	Chris

Table 2: Member Metrics for Sprint 1

Name	Points Attempted	Points Completed (Speed)	Work (hrs)	Cycle (hrs/pts)
Chris	5	5	10	2
Jack	10	8	14	1.75

6.1.1 Sprint 1 UML
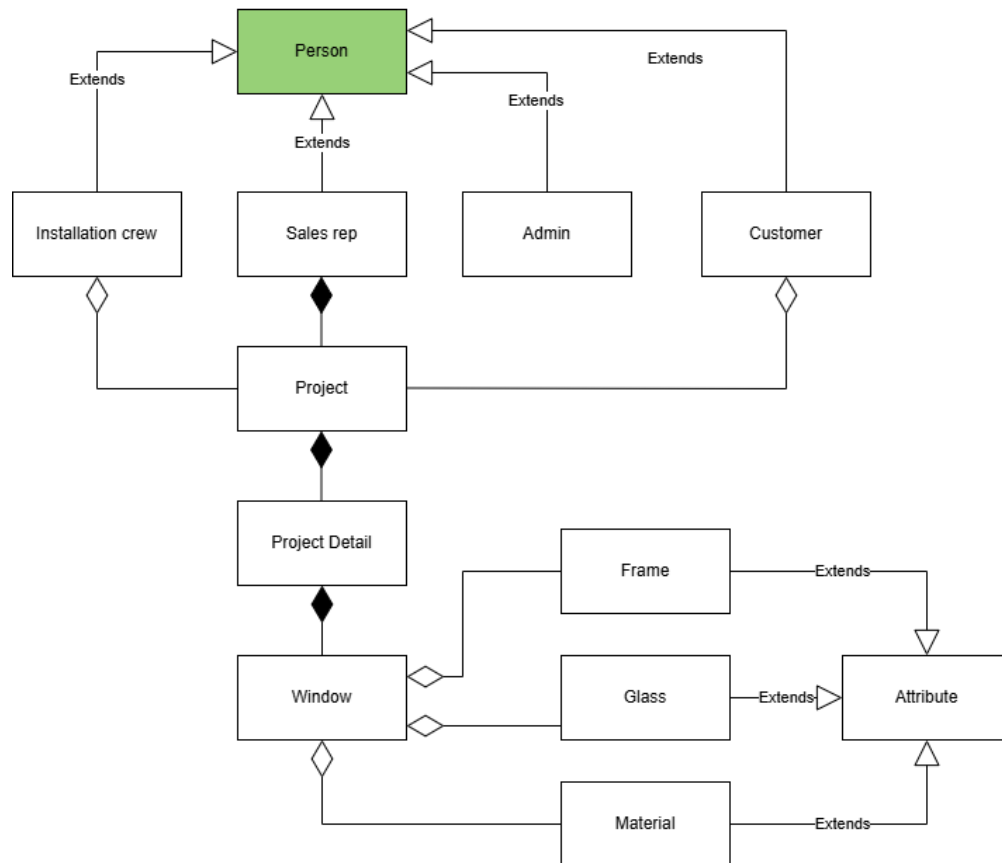
Figure 6: Updated UML: The green indicates what has been implemented this sprint.

In this updated UML diagram, we have highlighted the completed classes in green. During this sprint, we finished the Person class. With Django, classes like person are made as models and are contained in models.py. This interacts directly with the database, and since our ERD only contains a person entity, we may not need subclasses of Person in our future sprints.

6.1.2 Sprint 1 Testing

```
Name                                                                        Stmts   Miss  Cover   Missing
-----------------------------------------------------------------------------------------------------------
designer\__init__.py                                                            0      0   100%
designer\admin.py                                                               1      0   100%
designer\apps.py                                                                4      0   100%
designer\class_functions\__init__.py                                            0      0   100%
designer\class_functions\person.py                                             53     13    75%   53, 67-73, 80, 87-95
designer\migrations\0001_initial.py                                             5      0   100%
designer\migrations\0002_alter_person_department_alter_person_first_name_and_more.py    4      0   100%
designer\migrations\0003_alter_person_email.py                                  4      0   100%
designer\migrations\__init__.py                                                 0      0   100%
designer\models.py                                                              9      0   100%
designer\tests.py                                                              22      0   100%
designer\urls.py                                                                3      0   100%
designer\views.py                                                              31     25    19%   7, 10, 13-42
manage.py                                                                      11      2    82%   12-13
website\__init__.py                                                             0      0   100%
website\secrets.py                                                              5      0   100%
website\settings.py                                                            22      2    91%   79-80
website\urls.py                                                                 3      0   100%
-----------------------------------------------------------------------------------------------------------
TOTAL                                                                          177     42    76%
PS C:\Users\jackf\Documents\School Work\Capstone\capstone-formato-plowman\src\App\website>
```

Figure 7: Coverage report for sprint 1 code

6.1.3 Sprint 1 Retrospective & Reflection

This sprint was a good starting point for our project. We achieved the goals that we made for ourselves and are in a good position for later sprints. Learning PostgreSQL and Django was the most time consuming. They are very different than tools we have previously used, but they work great together and make the project easier. With Django, we may not need as many classes as we originally thought. This is because of how Django interacts with the database data. We will have to experiment as we continue through the project to understand what is best.

6.1.4 Next Iteration Planning

For sprint 2 we plan to polish up what we did in sprint 1 and then move to the CRUD operations for projects and details, as well as start to code some of the image processing functionality.

6.2 Sprint 2

Table 3: Overview for Sprint 2

ID	Name	Points	Contributor
S10	CRUD Project	8	Chris
S11	CRUD Project Detail	8	Jack
S2	Select Window	5	Chris
T3	Website Polishing	2	Chris
Spike 1	Automated Object Detection	N/A	Chris

This sprint, we had a spike that needed to be figured out before we could complete S2 (Select Window). This was to figure out how to use a deep learning Ai model and train it on a data set of windows on a house to accurately detect windows within the picture. We were not able to complete this spike or S2 by the end of Sprint 2, we did look into using Opencv as a tool which didnt work out due to it's inaccuracy.

Table 4: Metrics for Sprint 2

Name	Points Attempted	Points Completed (Speed)	Work (hrs)	Cycle (hrs/pts)
Chris	7	2	15	7.5
Jack	16	16	10	0.63

*Note that most of the time Chris spent was working on the Spike, trying to figure out how to train a deep learning Ai model.
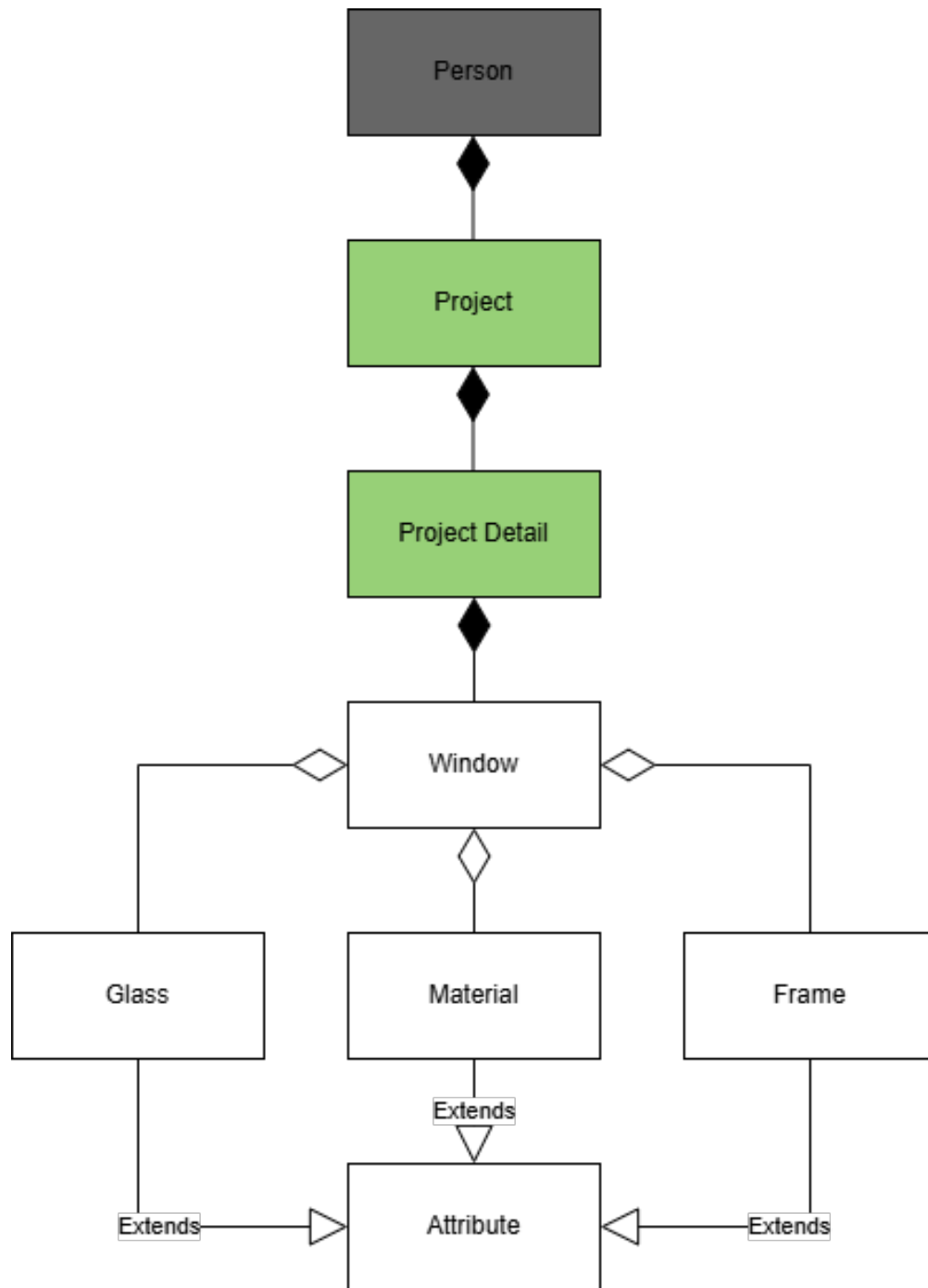
6.2.1 Sprint 2 UML

Figure 8: Updated UML: The green indicates what has been implemented this sprint. Gray is what has already been implemented.

After last sprint, we realized that our UML had unneeded classes. So, we updated it and removed all sub-classes of Person. Django's models connect directly to the database, so just having an attribute differentiate the type of person is enough.

6.2.2 Sprint 2 Testing

```
Name                                      Stmts   Miss  Cover   Missing
-----------------------------------------------------------------------
designer\__init__.py                          0      0   100%
designer\admin.py                             1      0   100%
designer\apps.py                              4      0   100%
designer\class_functions\__init__.py          0      0   100%
designer\class_functions\person.py           57     17    70%   43-44, 58-64, 70-71, 78-86
designer\class_functions\project.py          33      6    82%   8, 29-30, 34, 48-49
designer\class_functions\project_detail.py   32      5    84%   10, 30-31, 43-44
designer\migrations\0001_initial.py           6      0   100%
designer\migrations\__init__.py               0      0   100%
designer\models.py                           22      2    91%   15, 24
designer\tests.py                            84      0   100%
designer\urls.py                              3      0   100%
designer\views.py                            33     24    27%   8, 11, 14, 17, 21-54
manage.py                                    11      2    82%   12-13
website\__init__.py                           0      0   100%
website\secrets.py                            5      0   100%
website\settings.py                          24      2    92%   80-81
website\urls.py                               3      0   100%
-----------------------------------------------------------------------
TOTAL                                       318     58    82%
```

Figure 9: Coverage report for sprint 2 code

6.2.3 Sprint 2 Retrospective & Reflection

We got rid of S12 CRUD Account because it is redundant. The person (soon to be user) entity covers all account attributes and serves as the account.

Sprint 2 came with its challenges. All the CRUD operations were completed, but we struggled to get through Select Window. Chris spent most of his time researching frameworks and models to make this story work, but needed more time. Chris used the remainder of the sprint to polish the frontend instead, and plans on continuing this story in the next sprint.

6.2.4 Next Iteration Planning

In our next sprint we are focusing on getting the window design system functional. We plan to work on getting a deep learning Ai model trained to detect windows within our pictures. Additionally, we plan on refactoring our 'Person' table in our database to use Django's built in users table. If we can get those things done, then we plan on starting the process of making the design tool interactable.

6.3 Sprint 3

Table 5: Overview for Sprint 3

ID	Name	Points	Contributor
S2	Select Window	5	Chris
S4	Edit Border	2	Chris
S5	Manual Add	5	Chris
S6	Remove Window	2	Chris
S21	CRUD Window	8	Jack
Spike 1	Automated Object Detection	N/A	Jack & Chris

For half the sprint the two of us worked on the spike. We focused on using OpenCV's computer vision to detect windows automatically. Although it improved, we found that it wasn't feasible within our time frame. We then moved to manual, user driven system. The stories completed this sprint were centered around the manual version. Given this change, our iteration planning changed, we pushed some stories back and added a new one. The new story was S21: CRUD Window which was a needed story given our ERD. We had to push back S3: Customize Window, S7: Save Window, and S8: Save Project because they were dependent on other stories.

Table 6: Metrics for Sprint 3

Name	Points Attempted	Points Completed (Speed)	Work (hrs)	Cycle (hrs/pts)
Chris	14	14	22	0.64
Jack	8	8	20	0.4

6.3.1 Sprint 3 UML

Our UML was simplified this sprint. We transitioned from our Person model to built-in Django Users. Person was removed from the UML to account for this. After speaking with our client, we were told that previews of glass and different materials were unnecessary, so they were also removed from our UML. With only Frame remaining, the superclass Attribute was unnecessary. We are left with a simplified diagram consisting of four classes: Project, Project Detail, Window, and Frame.
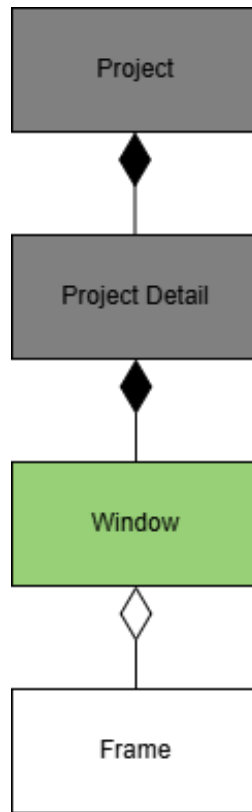
Figure 10: Updated UML: The green indicates what has been implemented this sprint. Gray is what has already been implemented.

6.3.2 Sprint 3 Testing

Figure 11: Coverage report for sprint 3 frontend code

*Note that the uncovered lines are all related to the creation/deletion of HTML elements based on user interaction.

```
Name                                        Stmts   Miss Branch BrPart  Cover   Missing
------------------------------------------------------------------------------------
designer\__init__.py                           0      0      0      0   100%
designer\admin.py                              1      0      0      0   100%
designer\apps.py                               4      0      0      0   100%
designer\class_functions\__init__.py           0      0      0      0   100%
designer\class_functions\project.py           33      0      4      0   100%
designer\class_functions\project_detail.py    32      0      2      0   100%
designer\class_functions\window.py            36      0      2      0   100%
designer\migrations\0001_initial.py            7      0      0      0   100%
designer\migrations\__init__.py                0      0      0      0   100%
designer\models.py                            20      1      0      0    95%   26
designer\tests.py                            132      0      0      0   100%
designer\urls.py                               3      0      0      0   100%
designer\views.py                             21     13      4      0    32%   8, 11, 14-27, 30
manage.py                                     11      2      2      1    77%   12-13, 21->exit
website\__init__.py                            0      0      0      0   100%
website\secrets.py                             5      0      0      0   100%
website\settings.py                           24      2      0      0    92%   80-81
website\urls.py                                3      0      0      0   100%
------------------------------------------------------------------------------------
TOTAL                                        332     18     14      1    93%
```

Figure 12: Coverage report for sprint 3 backend code

*Note that BrPart refers to partials, and 0 means that all branches are covered.

6.3.3 Sprint 3 Retrospective & Reflection

Sprint 3 came with hard decisions. The more time we spent on automatic detection, the more we realized that this goal was unattainable for the project and the less time we had for other features/stories. We decided to shift to a manual version of our project, which we managed to make significant progress on. Some stories had to be moved to later sprints, but this gap was filled by completing stories from future sprints. Along with these changes, we also refactored our Person model and transitioned to the built-in Django User model. This allowed for easy and secure login authentication.

6.3.4 Next Iteration Planning

Our client has emphasized the importance of the progress tracker component of our project. Based on this, we will be replanning our stories to accommodate a focus on the progress tracker. Unrelated to the progress of an automated version for the window design tool, they expressed less of an interest in that compared to the progress tracker, further changing our plans for the rest of the project.

6.4 Sprint 4

Table 7: Overview of Sprint 4

ID	Name	Points	Contributor
S3	Customize Window	8	Jack
S7	Save Window	2	Jack
S8	Save Project	2	Jack
S6	View Project Status	3	Chris
S14	Compact View	3	Chris
S?	CRUD Preview	8	Jack
S?	CRUD Customer	8	Jack
S?	CRUD Status	8	Jack
T4	Website Style Update	1	Chris

This sprint, we shifted our focus to the progress tracker component of our project per our client's request. Jack realized that the database needed some changes and ended up adding a few more CRUD operations to things like previews for projects, customers, and statuses for the progress tracker. Chris moved over to start working on the progress tracker and was able to get a simple version working, but has not yet implemented it with the rest of the system. We also improved the design of the website by replacing text headers with the company logo throughout the different pages.

Table 8: Metrics for Sprint 4

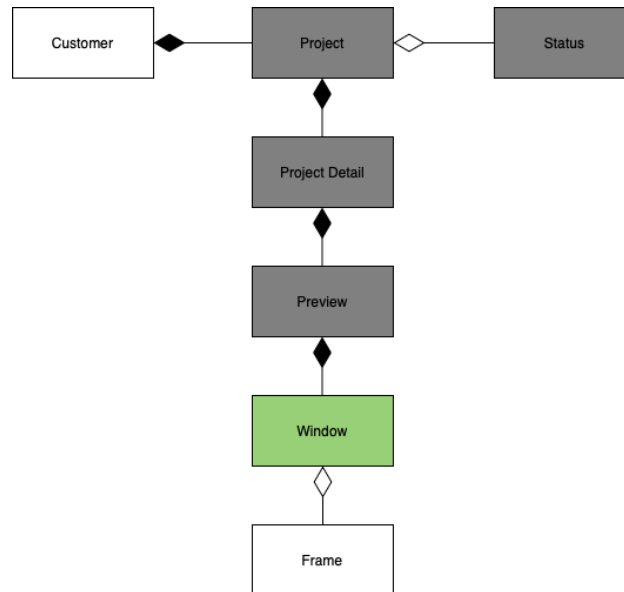| Name	Points Attempted	Points Completed (Speed)	Work (hrs)	Cycle (hrs/pts)
Chris	7	7	6	0.85
Jack	24	24	10	.42

6.4.1 Sprint 4 UML

Figure 13: Updated UML: The green indicates what has been implemented this sprint. Gray is what has already been implemented.

6.4.2 Sprint 4 Testing


```
Name                                     Stmts   Miss Branch BrPart  Cover   Missing
------------------------------------------------------------------------------------
designer/__init__.py                         0      0      0      0   100%
designer/admin.py                            1      0      0      0   100%
designer/apps.py                             4      0      0      0   100%
designer/class_functions/__init__.py         0      0      0      0   100%
designer/class_functions/customer.py        43     11      6      1    76%   11-12, 19-20, 35-36, 41, 45-46, 54-55
designer/class_functions/project.py         94     39     22      5    53%   17-18, 34-39, 48->55, 51, 56-59, 64-74, 89->88, 95-96
, 98->105, 108-127
designer/class_functions/status.py          32     10      0      0    69%   7-8, 15-16, 23-24, 29-30, 37-38
designer/migrations/0001_initial.py          7      0      0      0   100%
designer/migrations/__init__.py              0      0      0      0   100%
designer/models.py                          47      4      0      0    91%   13, 20, 32, 44
designer/tests/__init__.py                   0      0      0      0   100%
designer/tests/test_customer.py             38      4      0      0    89%   38-41
designer/tests/test_project.py              50      0      0      0   100%
designer/tests/test_status.py               22      0      0      0   100%
designer/urls.py                             3      0      0      0   100%
designer/views.py                           22     14      4      0    31%   8, 11, 14-28, 31
manage.py                                   11      2      2      1    77%   12-13, 21->exit
website/__init__.py                          0      0      0      0   100%
website/secrets.py                           5      0      0      0   100%
website/settings.py                         24      2      0      0    92%   80-81
website/urls.py                              3      0      0      0   100%
------------------------------------------------------------------------------------
TOTAL                                      406     86     34      7    76%
```

Figure 14: Coverage report for sprint 4 backend testing

6.4.3 Sprint 4 Retrospective & Reflection

Throughout our project, we have made multiple changes to our UML. Looking back, our planning should have been better. We added three more classes for this sprint alone, but now we are confident that there will be no more major changes to it. This should set us up for our final sprint.

6.4.4   Next Iteration Planning

In the next sprint, we plan on finishing the 2 most important components of the project. The window designer and the progress tracker. We will have to forgo some of the smaller aspects from the original plan due to lost time over an automated version of the window designer.