

C868 – Software Capstone Project Summary

Task 2 – Section A



Capstone Proposal Project Name: AppointmentKeeper

Student Name: Zachary Cabrera

Table of Contents

Contents

Table of Contents	2
Business Problem	3
The Customer	3
Business Case	3
Fulfillment	3
Existing Gaps	4
SDLC Methodology	4
Deliverables	5
Project Deliverables	5
Product Deliverables	5
Implementation	6
Validation and Verification	6
Environments and Costs	7
Programming Environment	7
Environment Costs	7
Human Resource Requirements	7
Human Resource Costs	8
Project Timeline	8

Business Problem

The Customer

Project1 is a company that provides project management and planning consultations to small or medium-sized businesses located within their geographic region. Project1 employees are full-time consultants responsible for capturing customer details and needs, scheduling appointments, and meeting with customers over multiple sessions to deliver a comprehensive consultation. Although historically an east-coast based company, Project1 will open their first sister branch on the west coast by the end of the calendar year. Project1's goal is to enable their customers to do more by streamlining project processes.

Business Case

Project1's business success is entirely dependent on interfacing with customers through appointments to deliver consultations. Therefore, it is imperative that systems are in place to accurately manage these two aspects: customers and appointments.

Project1 consultants each have a handful of customers they are working with at any given time and are responsible for capturing details for potential new customers. It's crucial for consultants to input accurate and complete customer information to be used as a reference during audits and for other business purposes such as using contact information for promotions. Issues with customer information such as missing or inaccurate data have negatively impacted the overall business. There have been instances of consultants unable to reach customers due to incomplete contact information. Additionally, incomplete customer data diminishes the effectiveness of business analytics reports. The proposed AppointmentKeeper application will solve these issues by presenting a single and consistent customer management interface to eliminate inconsistencies and gaps in customer information.

Accurate appointment scheduling is equally important. Project1 consultants manage busy schedules meeting with dozens of customers per day. There is a history of consultants erroneously double-booking customer appointments which has led to customer frustration and a loss of business in some extreme cases. Project1 management also needs to be informed of all appointments to have a good understanding of the current appointment load. This and other appointment metrics will help management make informed business decisions. The proposed AppointmentKeeper application will solve these issues by providing consultants with a consolidated view of appointments and integrated checks to prevent double-booking. AppointmentKeeper will also generate reports to meet management's appointment tracking needs.

Consultants do most of their work on the road. They jump between customer locations for appointments and need to manage customers and appointments when they are out of the office. AppointmentKeeper will leverage a cloud (Azure) MySQL database which will allow consultants to fully utilize the app, even when they are out of the office.

Fulfillment

AppointmentKeeper will fulfil the need for accurate customer record keeping by requiring consultants to enter the same information for each customer. Input validation within customer forms will ensure consistency when consultants add or edit customers. Consultants will be notified via error

Project1: AppointmentKeeper Solution

messages if there is missing or improperly formatted customer information. Consultants will also be able to search for customer names to quickly locate customer contact information.

AppointmentKeeper will enforce similar input validation and present error messaging when consultants add or edit appointments. Double-booking appointments and scheduling appointments outside of business hours will also be blocked by default. Consultants can choose the type of appointment from a preconfigured list and tie a specific customer to an appointment.

AppointmentKeeper will also generate reports for management such as displaying the total number of appointments by type per month, appointment schedule per consultant, and total appointments.

To facilitate consultants who are on the go, AppointmentKeeper will leverage an Azure MySQL database housing the appointment and customer records which can be accessed from anywhere via the internet.

Existing Gaps

Currently, Project1 uses a spreadsheet hosted on a shared server in the office to maintain customer records. The spreadsheet does not have any input validation to ensure consultants are entering valid information. Furthermore, consultants regularly omit certain customer information leading to incomplete records. Although management has stressed the importance of accurate record keeping, there is no mechanism to ensure their directives are enforced. A new system is needed to guarantee consistency in customer recordkeeping.

Additionally, new consultants hired at the west coast branch would need their own spreadsheet for record keeping. This is an undesirable solution as records would be split between two systems and could lead to additional inconsistencies in data. A centralized database is necessary that can be utilized by all branches.

Consultants manage appointments with traditional pen and paper in a daily planner. This manual process presents several issues. Firstly, management has no visibility into consultant's workloads. Without knowing consultant schedules, managers are unable to accurately allocate new customer leads to the consultants with bandwidth for additional work. A system is needed to allow managers to understand each consultant's workload to efficiently utilize their work hours and meet customer demand. Secondly, manual appointment scheduling is highly susceptible to error. There is history of consultants double booking appointments or booking appointments outside of business hours. These errors violate company policy, lead to missed or rescheduled appointments, and contribute to an unsatisfactory customer experience. A system is needed to prevent appointment scheduling conflicts. Lastly, consultants have missed appointments which leads to customer frustration and tarnishes Project1's professional reputation. A system is needed to remind consultants of upcoming appointments.

SDLC Methodology

The Waterfall SDLC Methodology has been chosen for this project. Project1 is a mature business with an excellent understanding of their business processes. Project1 also has a clear understanding of dependencies, timelines, and success metrics. These insights help define clear and detailed project

Project1: AppointmentKeeper Solution

requirements at the onset, which lends to the strengths of the Waterfall method. Additionally, Project1 stakeholders are preoccupied with the west coast branch opening and have little time to dedicate to this project's development. The Waterfall method is ideal for this circumstance because it frontloads Project1 stakeholder involvement and is not reliant on continuous stakeholder feedback like other SDLC methods. Lastly, Project1 is unfamiliar with SDLC methods and will appreciate the simplicity of linear Waterfall method steps.

The following phases are included in this Waterfall implementation:

- Planning: Requires the most Project1 stakeholder involvement to understand and document the business case, existing gaps, costs, risks, dependencies, success criteria, and requirements.
- Design: Develop logical design artifacts such as class and ERD diagrams. Develop a wireframe.
- Implementation: Produce the code to satisfy project requirements.
- Verification and testing: Debug code and perform other quality assurance tasks such as unit testing. Verify with Project1 stakeholders that the application meets all requirements.
- Maintenance and Operations: Deploy the application to production. Correct any errors or defects that arise during real-world use. Provide documentation such as user guides.

Deliverables

Project1 has requested specific results and outputs from this project. These deliverables are highlighted below in per the Waterfall SDLC.

Project Deliverables

This consists of items that are necessary for the project to proceed efficiently and smoothly to achieve the end goal of producing an application.

- Project Requirements Document
 - Minimum criteria must be met to consider the application complete.
- Project Timeline
 - Provides a breakdown of the project into the Waterfall phases with completion dates.
- Test Plans
 - A document that describes the testing strategy, objectives, and resources required to validate the product.
- Project Cost Estimation
 - Provides an estimate of costs for the completed solution. Project1 stakeholders sign off and approve the budget for the project.
- Status Reports
 - Weekly status reports illustrating how the project work is progressing.

Product Deliverables

This consists of the components to be delivered to the customer that directly pertain to the end product.

- Class Diagram
 - A diagram showing the logical structure of the application including classes, attributes, and methods.

Project1: AppointmentKeeper Solution

- Entity Relationship Diagram (ERD)
 - A diagram showing the relationships between entities in the MySQL database used by the application.
- Wireframes
 - A low fidelity representation of the application structure to convey the overall vision.
- User Flow Diagram
 - A visualization of how users will interact with the application and transition through actions.
- User Documentation
 - Includes a user guide on how to set up, use, and maintain the application

Implementation

Before the application rollout, Project1 management and senior consultants will participate in a closed beta wherein validation and verification will take place. When beta testing is complete and all requirements have been satisfied, production rollout will begin.

The AppointmentKeeper application will move to production in a phased approach with three stages. At launch, 1/3 of consultants will be onboarded to the application. This approach will limit the blast radius of potential business disruption due to a process change as well as build confidence in the stability of the application as the load on the database increases. The full rollout will take approximately three weeks with 1/3 of consultants onboarded at the beginning of each week. Project1 management is tasked with organizing consultants into the three rollout groups and providing this information to the application team to generate welcome emails for individuals at the appropriate time.

The AppointmentKeeper application final build will be housed in an Azure Blob Storage Account. Consultants will be distributed a user setup guide (.pdf file) via email with steps to onboard including a link to the Storage Account where they can download the application setup files. Consultant's unique usernames and passwords will also be distributed in the welcome email. Upskilling Project1 IT personnel on the Microsoft Azure components of this application is not included within the scope of this project. The application team will formally hand over all cloud assets to the Project1 IT staff two weeks after the completion of the rollout.

A change-control or maintenance window will not be necessary for this application rollout. However, consultants should allocate three hours to complete application onboarding and import manual records.

Validation and Verification

The application team is responsible for completing manual testing to ensure code is running error-free and the functionality meets application requirements. A flowgraph is used to identify all possible paths and the application team will write test cases to ensure all paths have been verified.

After the application team has completed testing, they will work with Project1 stakeholders to build user acceptance test cases that reflect the defined requirements. Examples of test cases include creating a new customer, editing an appointment, and generating reports. Prior to release, Project1 management and senior consultants will participate in a closed beta where they will perform black-box

user acceptance testing using the test cases. This test period will last two days and will conclude with a three-hour meeting to discuss the results. At validation completion, Project1 stakeholders will sign off on the application, confirming it meets all project requirements. Sign-off indicates the application is approved for formal rollout.

Environments and Costs

Programming Environment

- Development Environment
 - Windows 11 Home
 - Visual Studio 2019
 - MySQL Data package
 - Microsoft.Storage.Blobs package
 - Github account
- Database Environment
 - Microsoft Azure MySQL Database
- Application Distribution Environment
 - Microsoft Azure Storage Account

Environment Costs

This project includes startup and operational costs. Project1 currently has laptops for half of their consultants. As the solution is a desktop application, Project1 will need to source laptops for all consultants to fully benefit from this application. Hardware sourcing and acquisition are out of scope for this project. Operational costs cover the Microsoft Azure Cloud components of this project. The Azure MySQL Database – Single Server will run on the Basic SKU with monthly recurring costs of around \$28 and Service Level Agreement (SLA) of 99.99% uptime. As Project1 grows, there is a possibility that additional database resources will need to scale up or out which can increase costs in the future. Azure Storage Account guarantees 99.9% uptime SLA and will cost around \$2 per month. Data transfer from Azure also has a fee. However, the first 100GB of data transfer is free. This application is not expected to transfer over 100GB of data.

Human Resource Requirements

This project requires a Project Manager and a Development Team consisting of three C# developers and an Azure specialist.

The Project Manager is the primary point of contact for Project1 stakeholders and is responsible for delivering all project deliverables as well as defining deliverables for the project development team. The Project Manager will take ownership of defining the project phases and is expected to consume 20% of the project hours and budget.

One member of the Development Team will be designated as the Development Lead who is responsible for distilling requirements from the Project Manager into assignable tasks. The Development Lead communicates progress and possible complications with the Project Manager and completes development tasks. Application development by the Development Team is expected to consume 70% of the project hours and budget and will include developing the app and creating test cases. The remaining

Project1: AppointmentKeeper Solution

10% of the time and budget is allocated to the Azure Engineer to deploy, configure, and test the Azure environment.

Project1 must supply an IT team responsible for taking ownership of the application and Azure components after the formal handover.

Human Resource Costs

Resource	Hourly Rate	Hours	Total
Project Manager	28	15	\$420
Development Lead	30	18	\$540
Development Team	28	36	\$1,008
Azure Engineer	23	6	\$138
	Total:	75	\$2,106

Project Timeline

Phase	Deliverables	Description	Dates
Planning	Requirements Project Timeline Cost Estimate Test Plan	Several meetings with customer to understand requirements, set expectations, and sign off on project start and budget.	5/1/2022 – 5/2/2022
Design	Low fidelity wireframe Class Diagram ERD User Flow Diagram	Create the UI that relates the look and feel of the project. Create logical structures for app and database.	5/3/2022
Implementation	Application Code MySQL Database Azure Storage Account	Develop the application. Deploy cloud infrastructure.	5/4/2022- 5/5/2022
Verification and Testing	Production ready application	Complete manual testing and user black-box testing. Complete customer beta testing. Receive stakeholder project signoff.	5/6/2022
Maintenance and Operations	User documentation	Deploy to production. Educate users on proper use.	5/7/2022

C868 – Software Capstone Project Summary

Task 2 – Section C



Capstone Proposal Project Name: AppointmentKeeper

Student Name: Zachary Cabrera

[Table of Contents](#)

Contents

Table of Contents	10
Application Design and Testing.....	12
Design Document.....	12
Class Diagrams	12
Helper Classes	12
Entity Classes	13
UI Classes	14
Low Fidelity Wireframe.....	14
User Flow Diagram	17
Entity Relationship Diagram (ERD).....	17
Unit Test Plan	19
Overview	19
Test 1 – User authentication testing.....	19
Test components	19
Test Case	19
Outcome	19
Test Steps and Pass/Fail Criteria.....	19
Pass/Fail	20
Sample Test.....	20
Results.....	21
Test 2 – Add customer phone number input validation	21
Test components	21
Test Case	21
Outcome	21
Test Steps.....	21
Pass/Fail	22

Project1: AppointmentKeeper Solution

Sample Test.....	22
Results.....	23
User Guide	24
Introduction	24
Installation	24
How To: Login	25
Main Page Overview	26
How To: Add or Edit an Appointment.....	27
How To: Add or Edit a Customer.....	28
How To: Search for a Customer	28
How To: Use Reports	29
Administrator Guide	30
Introduction	30
MySQL Workbench setup	30
How To: Add new Users.....	30
How To: Delete Users	31
How To: Review Access Logs.....	33

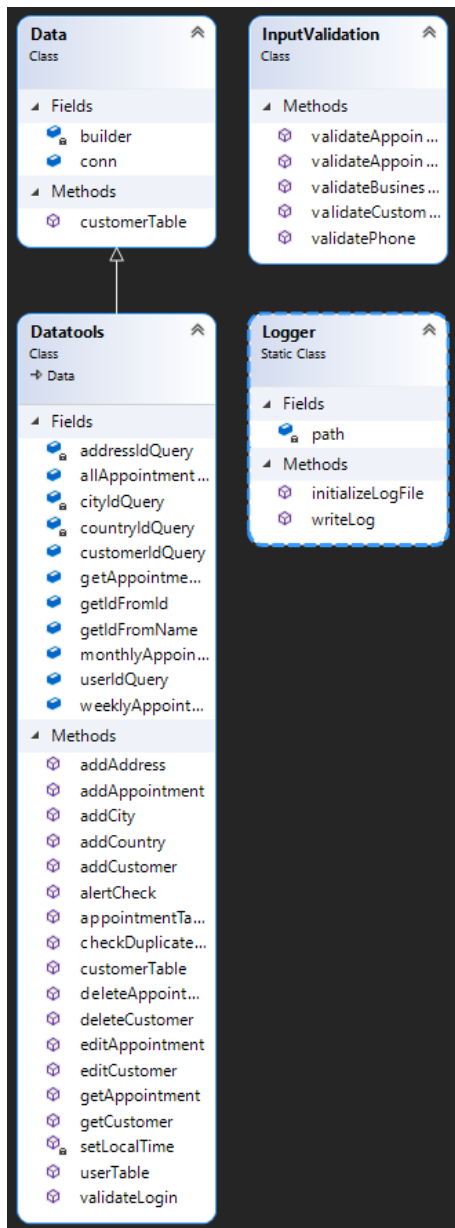
Application Design and Testing

Design Document

Class Diagrams

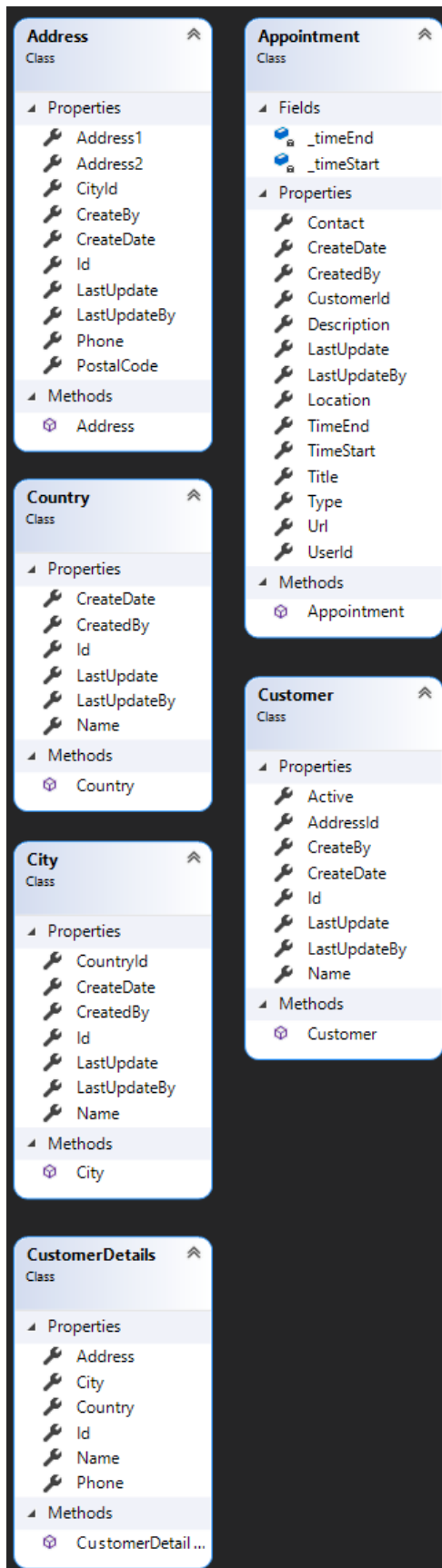
The class diagrams below describe the logical structure of the application including classes, their attributes or properties, and methods. Class diagrams are separated into three categories:

- Helper Classes
- Entity Classes
- UI Classes



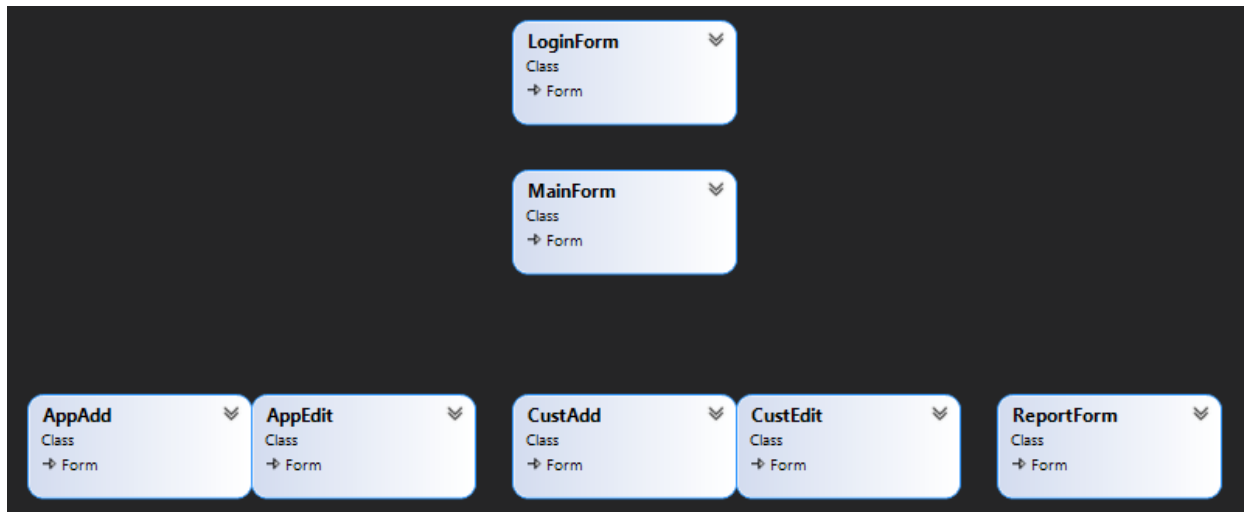
Helper Classes

The image to the left shows the Helper Classes of the application. The purpose of these classes is to consolidate common functionality needed by various aspects of the program into individual classes to promote code re-use. Data classes interface with the MySQL database and are used for retrieving, editing, and deleting records in the database. The InputValidation class holds all the methods to validate various input types across the application. Lastly, the Logger class handles user authentication logging.



Entity Classes

The image to the left shows the Entity Classes of the application. The entity classes represent various tables in the database which need to be regularly worked within the application. To promote clean and readable code, these classes were created to instantiate objects from the database and easily work with them in various parts of the application without requiring constant database calls.



UI Classes

The image above shows the UI Classes of the application. UI classes are derived from the Forms class and represent the UI components of the application. Each UI class equates to a unique UI element or form. Classes contain all the properties and constructors to render GUI elements as well as methods to interact with the application such as navigation.

Low Fidelity Wireframe

The low fidelity wireframes images below represent a basic blueprint of each form that will be included in the application. Wireframes will be presented to Project1 during the design phase. Project1 stakeholder approval and sign-off is required to progress to the implementation phase.

The wireframe shows a window titled 'Login' with a close button (X) in the top right corner. Inside the window, there are two input fields: 'Username' and 'Password'. Below these fields are two buttons: 'Cancel' on the left and 'Login' on the right.

Figure 1: Login Form

Project1: AppointmentKeeper Solution

The Main Form / Home window contains the following elements:

- Appointment Data**: A large rectangular area with a diagonal line from the bottom-left to the top-right.
- Customer Data**: A large rectangular area with a diagonal line from the bottom-left to the top-right.
- Buttons**:
 - Top right: Add, Edit, Delete
 - Below filters: Add, Edit, Delete
 - Bottom left: Exit
 - Bottom right: Reports
- Filters**: A box containing Filter1, Filter2, and Filter3.
- Search/Reset**: A search input field with Search and Reset buttons.

Figure 2: Main Form (Home page)

The Add/Edit Customer Form contains the following elements:

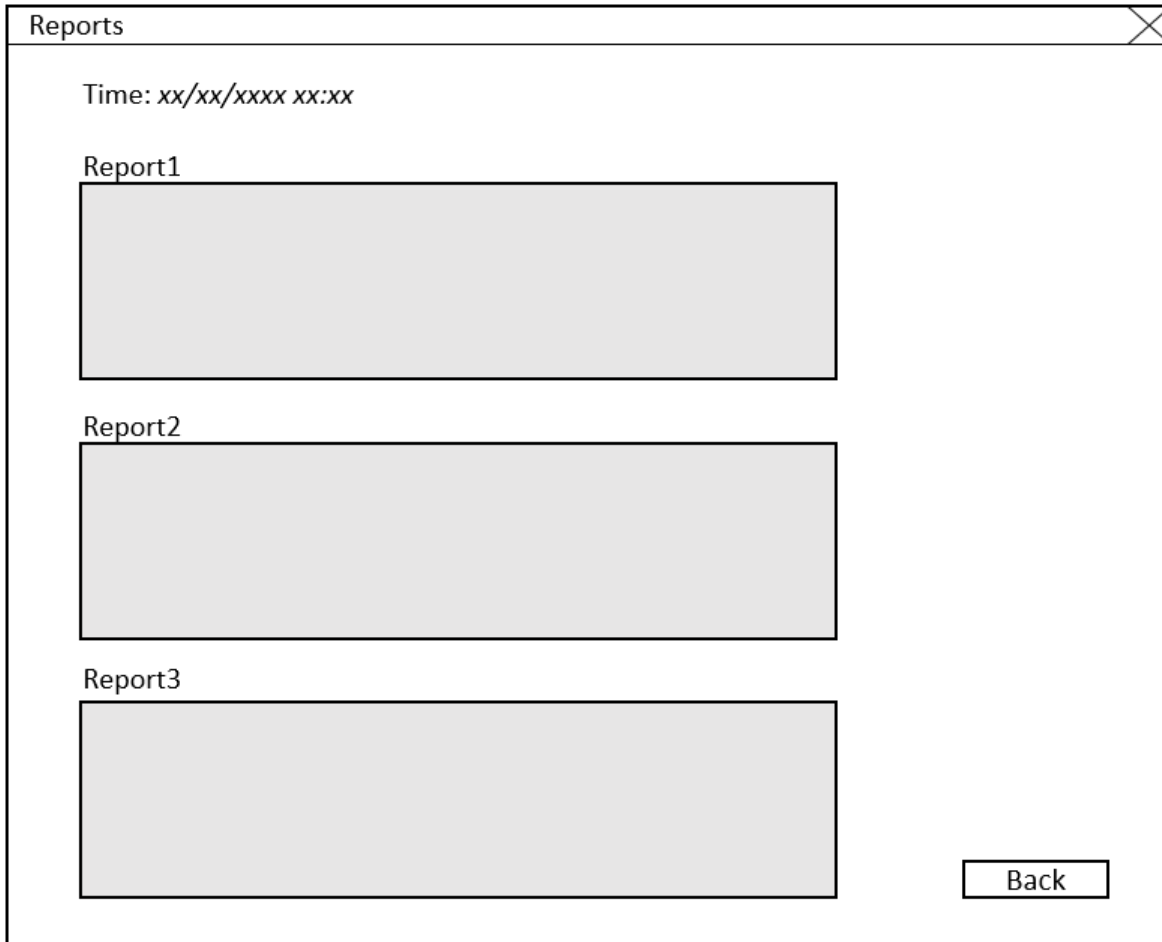
- Data1**, **Data2**, **Data3**, **Data4**, **Data5**: Five text input fields.
- Buttons**: Cancel and Save.

Figure 3: Add or Edit Customer Form

The Add/Edit Appointment Form contains the following elements:

- Data1**, **Data2**, **Data3**, **Data4**, **Data5**: Five dropdown menus.
- Buttons**: Cancel and Save.

Figure 4: Add or Edit Appointment Form

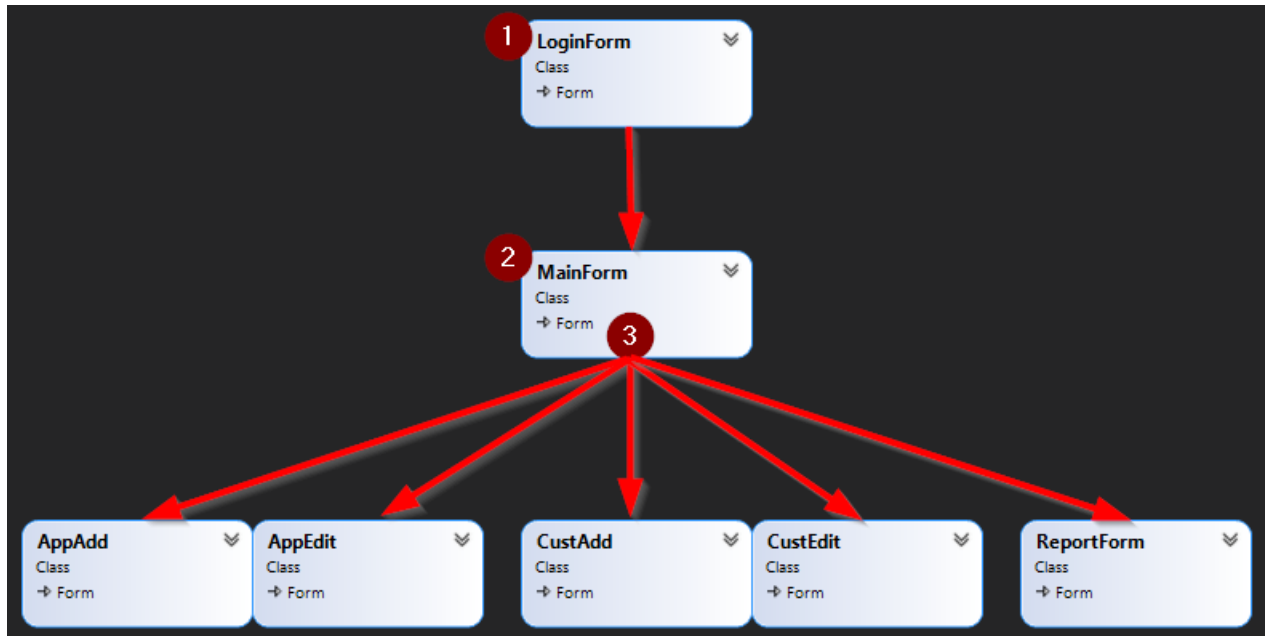


The image shows a window titled "Reports" with a close button in the top right corner. Inside the window, there is a time display "Time: xx/xx/xxxx xx:xx". Below this, there are three sections labeled "Report1", "Report2", and "Report3", each followed by a large, empty rectangular box for text input. In the bottom right corner of the window, there is a button labeled "Back".

Figure 5: Reports Form

User Flow Diagram

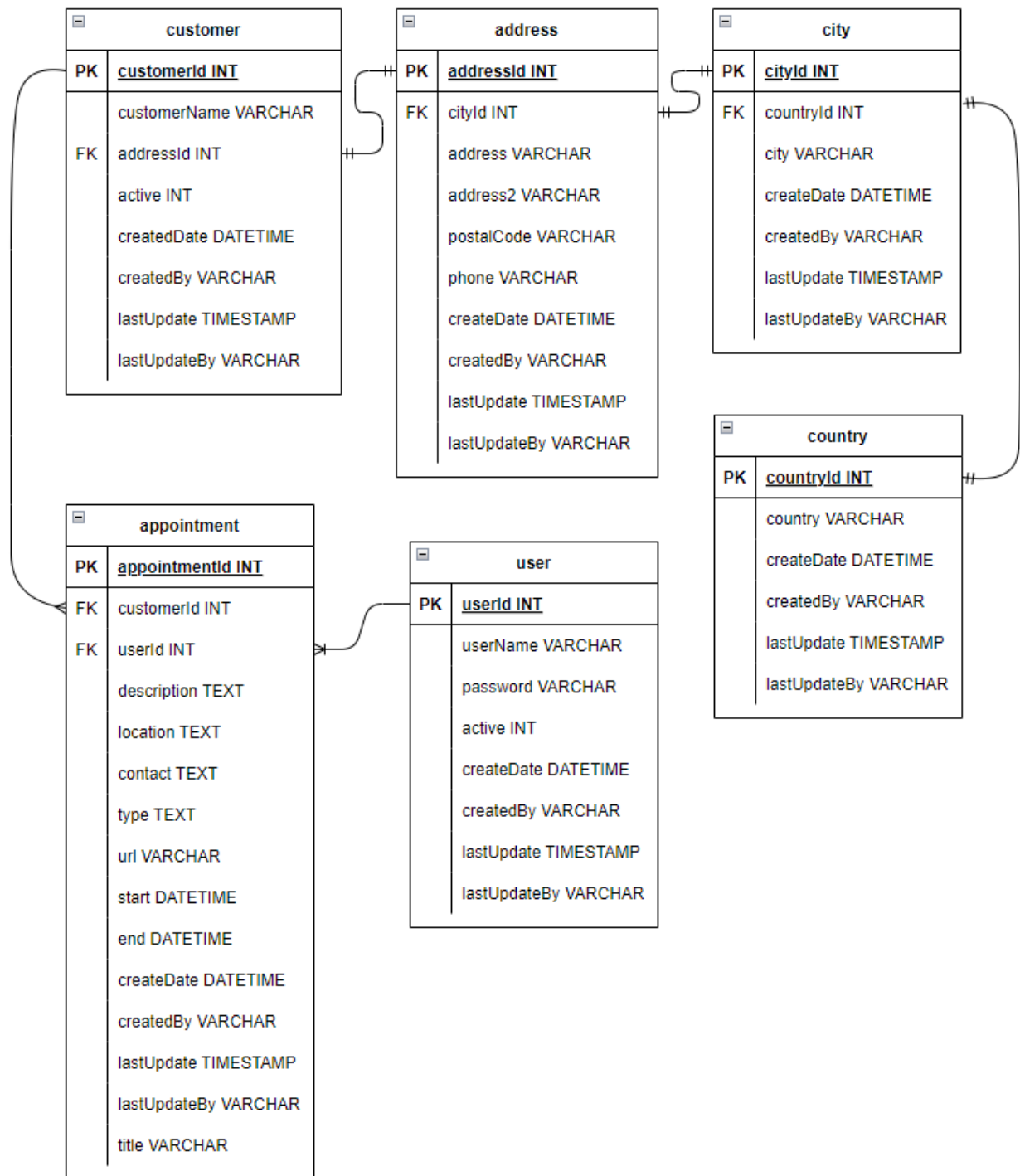
The user flow diagram depicts how a user will transition through the GUI elements of the application. In the previous section, the low fidelity wireframe showed each form in detail. In the flow diagram below, each GUI element is represented as a class like the class diagram, with the addition of arrows to show the path of the user. The user starts at the login page (1). After a successful login, the user will see the main page (2). From the main page, users can access additional pages to add or alter data or generate reports (3). Users are returned to the main page (2) after completing or terminating actions on third-level pages.



Entity Relationship Diagram (ERD)

The Entity Relationship Diagram below shows how entities or objects relate to each other within the MySQL Database used as the backend for this application. A customer has a one-to-one chained relationship with address, city, and country. This decision was made to reduce overall development time as there will be no concern for foreign key constraints when working with data. Both users and customers have a one-to-many relationship with appointments.

Project1: AppointmentKeeper Solution



Unit Test Plan

Overview

Manual application testing will be used to identify bugs, issues, and unexpected behavior in the application. Testing will confirm that application code, such as input validation and user authentication, are fully functional and producing expected results. If unexpected results are experienced during testing, developers will log the behavior, perform a code review, implement remediation, and retest. Any logged issues will be marked as complete once the code change has been verified to resolve the issue.

Test 1 – User authentication testing

Test components

To perform this test, the Azure MySQL database must be deployed and fully functioning. The user table in the database must contain at least one user with a username and password. The Azure Engineer or Application Team must provide the tester with a valid username and password. The tester must have internet connectivity and the completed application executable on their local Windows system.

Test Case

Upon submitting the login form, the application must query the database to confirm if the user has entered a username and password that matches an existing user record. If the user's credentials exist in the database, they will pass the login form and move to the main application form. If the credentials are invalid or do not exist in the database, the user will receive a message window informing them of the failed login.

Outcome

This test will confirm that the login form is fully functional and ready for user testing.

Test Steps and Pass/Fail Criteria

1. Make a note of the valid username and password. Example:
Username: test
Password: test
2. On a Windows system, launch the application executable. A login form is presented.
3. After each test below is completed, close and re-launch the application.
4. Test 1-A: Invalid username
 - a. Enter an invalid username such as "abc" and a valid password such as "test"
 - b. Pass: A message window shows the username or password is invalid.
 - c. Fail: The user is granted access to the main application page.

Project1: AppointmentKeeper Solution

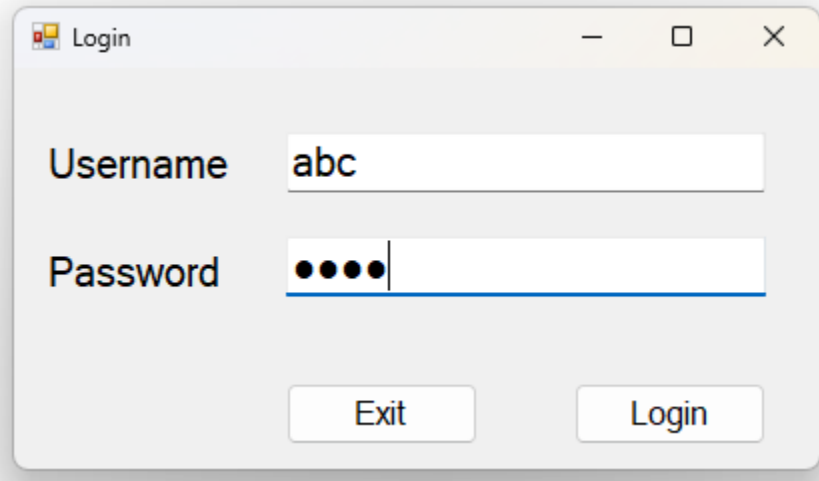
5. Test 1-B: Invalid password
 - a. Enter a valid username “test” and an invalid password such as “abc”
 - b. Pass: A message window shows the username or password is invalid.
 - c. Fail: The user is granted access to the main application page.
6. Test 1-C: Invalid username and password
 - a. Enter an invalid username such as “abc” and an invalid password such as “test”
 - b. Pass: A message window shows the username or password is invalid.
 - c. Fail: The user is granted access to the main application page.
7. Test 1-D: Valid username and password
 - a. Enter a valid username “test” and a valid password “test”
 - b. Pass: The user is granted access to the main application page.
 - c. Fail: The user is granted access to the main application page.

Pass/Fail

If tests 1-A through 1-D have passed, this Test 1 is considered passed. A failure in any sub-test constitutes a failure of Test1 and requires remediation.

Sample Test

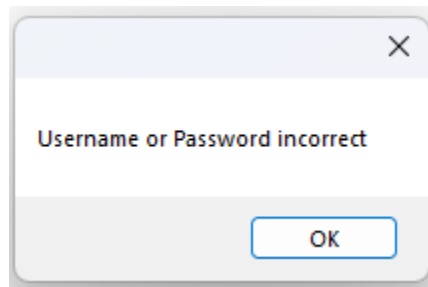
The screenshot below shows testing Test1-C (see Test Steps section above).



The screenshot shows a Windows-style login dialog box titled "Login". It contains two input fields: "Username" and "Password". The "Username" field contains the text "abc". The "Password" field contains four black dots, indicating a masked password. Below the input fields are two buttons: "Exit" and "Login". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

Results

The screenshot below shows the result of Test1-C. Pass criteria are for the message window to appear with the appropriate error. As the message window appeared, 1-C is considered passed.



Test 2 – Add customer phone number input validation

Test components

To perform this test, the Azure MySQL database must be deployed and fully functioning. The tester must have internet connectivity and the completed application executable on their local Windows system. The Azure Engineer must supply the tester with a valid username and password. The customer must not contain any record with customerName "a" and password "a".

Test Case

One of the goals of the application is to standardize customer records including customer phone numbers. Manual testing will confirm that user input for a customer phone number adheres to the standard format of 10 dash-separated digits. A properly formatted example number would be "111-111-1111". The application should accept properly formatted phone numbers and prevent improperly formatted numbers from saving to the database.

Outcome

This test will confirm that the input validation for customer phone numbers is fully functional and ready for user testing.

Test Steps

1. On a Windows system, launch the application executable. Enter the username and password provided by the Azure Engineer to authenticate to the application.
2. On the main page, click the "Add" button next to the customer table.
3. Enter "a" into all textboxes except the phone number text box.
4. Testing values:

Project1: AppointmentKeeper Solution

2-A: 1111111111
2-B: 111-1111111
2-C: 111111-1111
2-D: aaaaaaaaaa
2-E: aaa-aaa-aaaa
2-F: 111-111-111a
2-G: (111)111-1111
2-H: 111-111-1111

5. Perform the following steps and repeat for each value from 2-A through 2-H:
 - a. Enter the phone number.
 - b. Select the “Save” button.
 - c. Record the result in a log.

Pass/Fail

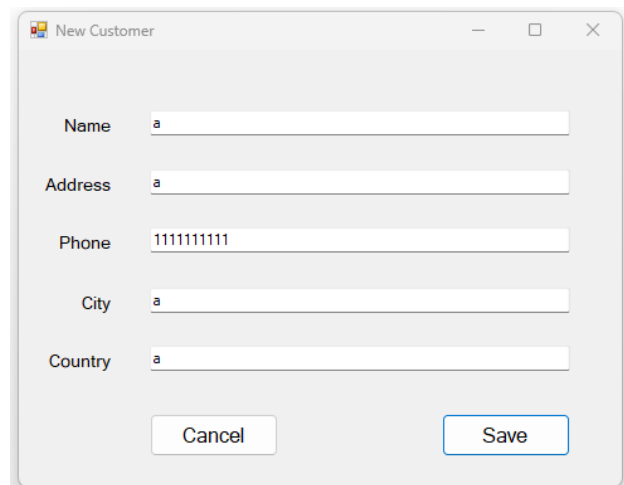
For this test to be considered passed, both criteria below must be true:

- Test 2-A through 2-G must present the error message window stating the phone number format is incorrect.
- Test 2-H must save the customer to the database without presenting any error message.

Any combination of results other than defined in the pass criteria above signifies a failure of this test and necessitates remediation.

Sample Test

The screenshot below shows testing Test2-A (see Test Steps section above).



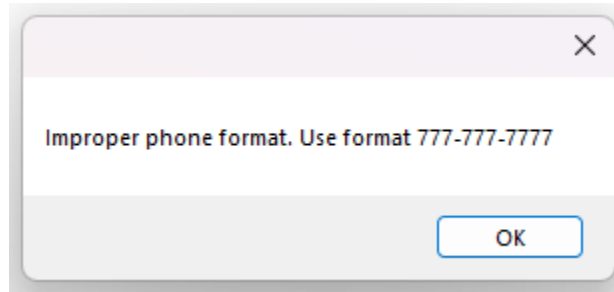
The screenshot shows a 'New Customer' dialog box with the following fields and values:

Field	Value
Name	a
Address	a
Phone	1111111111
City	a
Country	a

Buttons: Cancel, Save

Results

The screenshot below shows the result of Test2-1. Pass criteria are for the message window to appear with the appropriate error. As the message window appeared, 2-A is considered passed.



User Guide

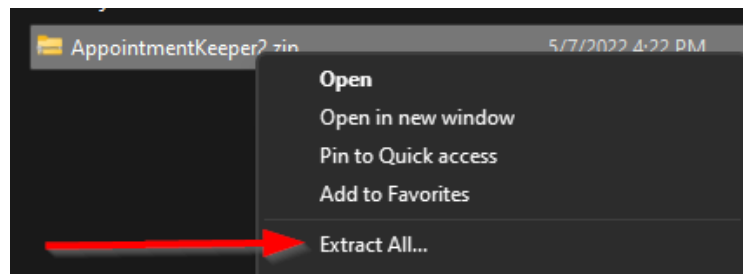
Introduction

Welcome to the AppointmentKeeper user guide! This guide will help users become familiar with the AppointmentKeeper interface and give a walkthrough of how to perform common tasks.

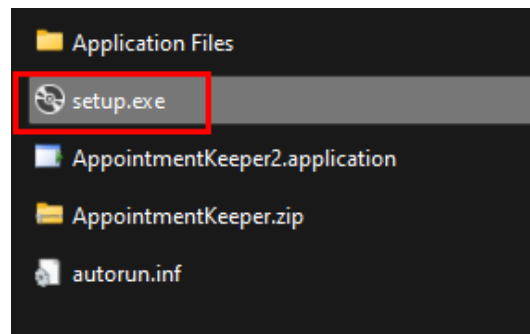
IMPORTANT NOTE: To use AppointmentKeeper, a username and password are necessary to access the user account. Contact an IT administrator to receive a username and password.

Installation

1. [Download AppointmentKeeper here](#) (an internet connection is required).
2. Locate the downloaded AppointmentKeeper2.zip file, right-click, and select Extract All.



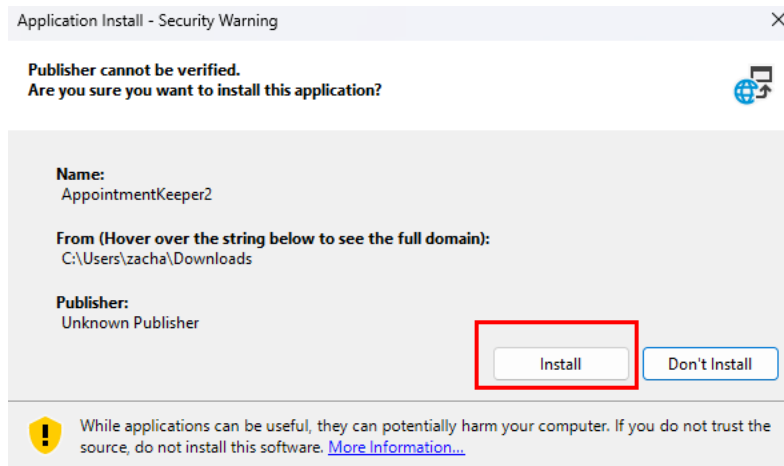
3. Once extracted, double click setup.exe. Select "Install".



TIP! After installation is complete, open Windows search by pressing the Windows key and type "AppointmentKeeper". When AppointmentKeeper2 appears, right-click, and select "Pin to taskbar". This makes opening AppointmentKeeper2 easy.

Project1: AppointmentKeeper Solution

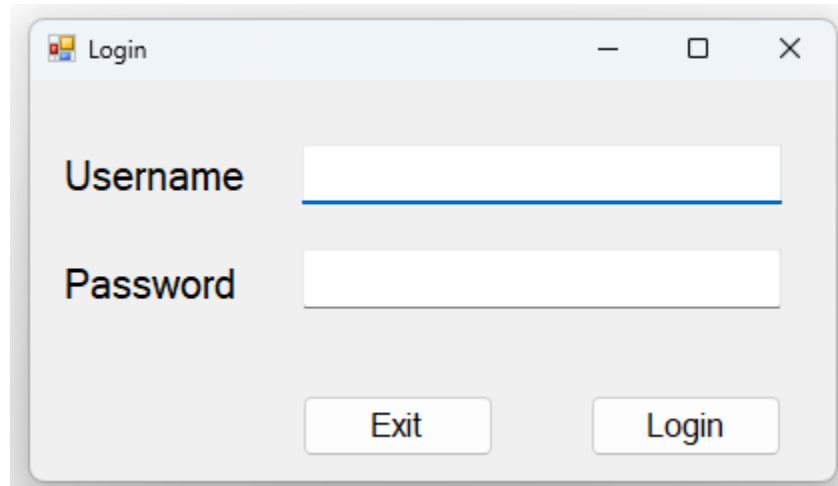
4. Select "Install".



5. Congratulations, AppointmentKeeper is installed!

How To: Login

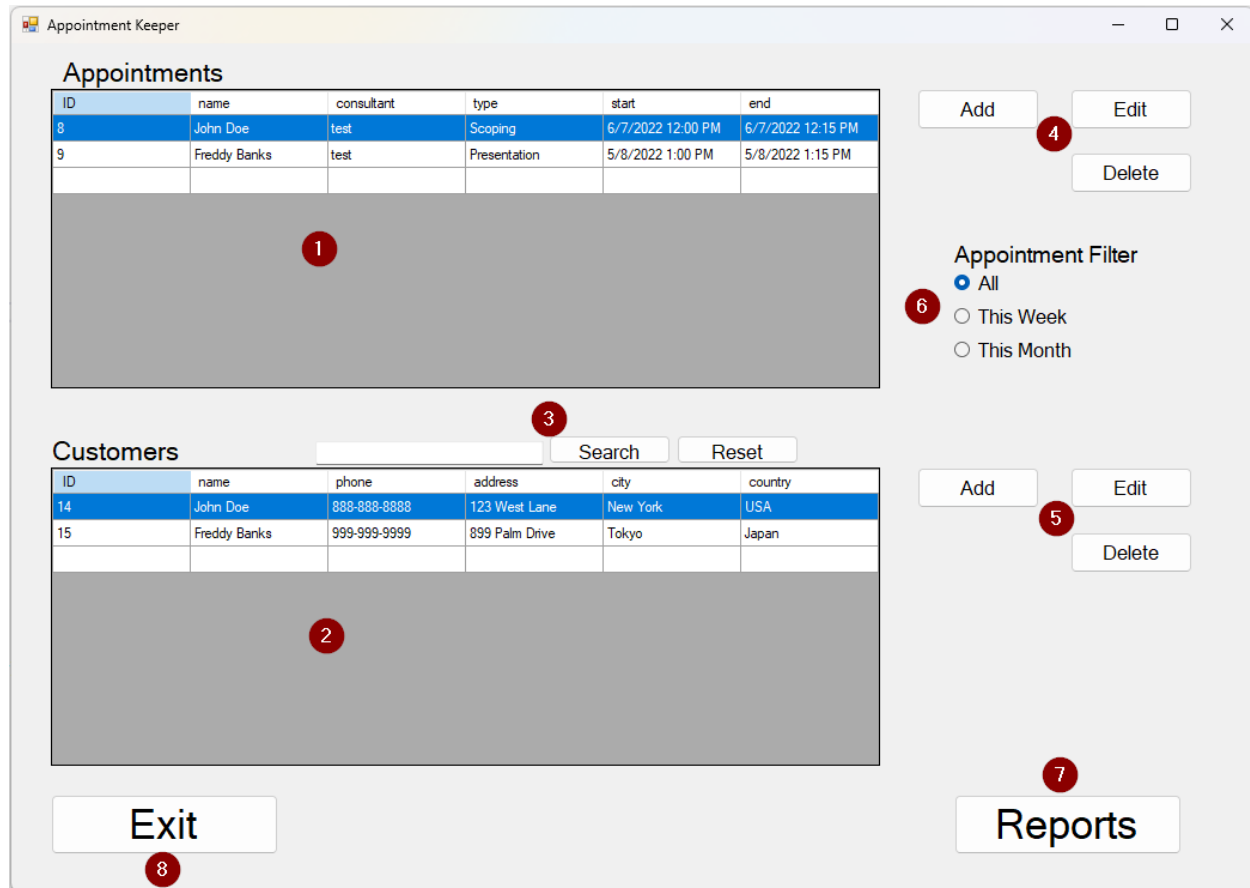
1. When first launching AppointmentKeeper, a login screen is presented:



Enter the username and password supplied by the administrator. If the username or password is incorrect, please verify both and try again. Reach out to the administrator for username and password-related issues.

Main Page Overview

Review the image below to become familiar with the layout of AppointmentKeeper.

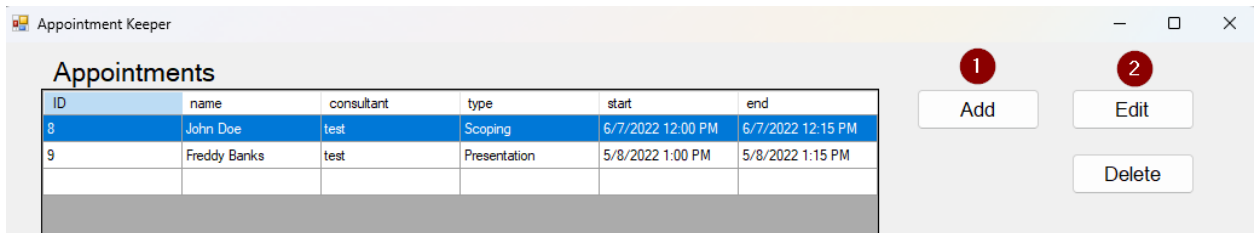


1. Appointments schedule: Here you can find all the scheduled appointments. Click the column names to sort by each column. For example, select the "name" column to sort appointments alphabetically by the customer's name.
2. Customer table: Here you can find all the customers entered into the system. Click the column names to sort by each column. For example, select the "name" column to sort customers alphabetically by their name.
3. Search Box: Enter a customer name into the search box and select "Search" to check if a customer exists with that name. When you are finished searching, select "Reset" to show all customers.
4. Add, Edit, and Delete Appointments.
5. Add, Edit, and Delete Customers.
6. Appointment Filter: Select a filter button to filter the appointments table to show all appointments or appointments for this week or month.
7. Reports: Select the Reports button to open the reports window.
8. Exit: Select Exit to close the application.

Project1: AppointmentKeeper Solution

How To: Add or Edit an Appointment

1. Login to Appointment Keeper.
2. If you would like to add a new Appointment, select Add (1). If you would like to edit an existing appointment, select the appointment in the appointment table and select Edit (2)



3. In the Add or Edit appointment screen, ensure all boxes are filled with information.

TIP! You cannot set an appointment outside of business hours or in the past. Additionally, appointments cannot overlap with other appointments.

4. Click Save (1) when everything is filled out.

The screenshot shows the 'New Appointment' dialog box. It contains the following fields and controls:

- Type: Scrum (dropdown menu)
- Customer: John Doe (dropdown menu)
- Time: 05/07/2022 17:06 (datetime picker)
- Duration: 15 minutes (dropdown menu)
- Consultant: test (dropdown menu)
- Buttons: Cancel and Save (the Save button is marked with a red circle 1)

Project1: AppointmentKeeper Solution

How To: Add or Edit a Customer

1. Login to Appointment Keeper.
2. If you would like to add a new customer, select Add (1). If you would like to edit an existing customer, select the customer in the customer table and select Edit (2)

The screenshot shows a web interface for managing customers. At the top, there is a 'Customers' header, a search text box, and 'Search' and 'Reset' buttons. Below this is a table with columns: ID, name, phone, address, city, and country. The table contains two rows: one for 'John Doe' (ID 14, phone 888-888-8888, address 123 West Lane, city New York, country USA) and one for 'Freddy Banks' (ID 15, phone 999-999-9999, address 899 Palm Drive, city Tokyo, country Japan). To the right of the table are three buttons: 'Add' (marked with a red circle 1), 'Edit' (marked with a red circle 2), and 'Delete'.

ID	name	phone	address	city	country
14	John Doe	888-888-8888	123 West Lane	New York	USA
15	Freddy Banks	999-999-9999	899 Palm Drive	Tokyo	Japan

3. In the Add or Edit appointment screen, ensure all boxes are filled with information.
TIP! You cannot create duplicate customers with the same details.
4. Click Save (1) when everything is filled out.

The screenshot shows a dialog box titled 'Edit Customer'. It contains five text input fields: 'Name' (with 'John Doe' entered), 'Address' (with '123 West Lane' entered), 'Phone' (with '888-888-8888' entered), 'City' (with 'New York' entered), and 'Country' (with 'USA' entered). At the bottom of the dialog are two buttons: 'Cancel' and 'Save' (marked with a red circle 1).

How To: Search for a Customer

1. Login to Appointment Keeper.
2. Enter the customer's name into the search text box (1). Click Search (2).

The screenshot shows the same 'Customers' table interface as before. The search text box now contains the text 'John' (marked with a red circle 1). The 'Search' button (marked with a red circle 2) is highlighted. The 'Reset' button (marked with a red circle 3) is also visible. The table content remains the same.

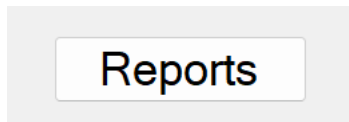
ID	name	phone	address	city	country
14	John Doe	888-888-8888	123 West Lane	New York	USA

TIP! Enter any part of a customer's name.

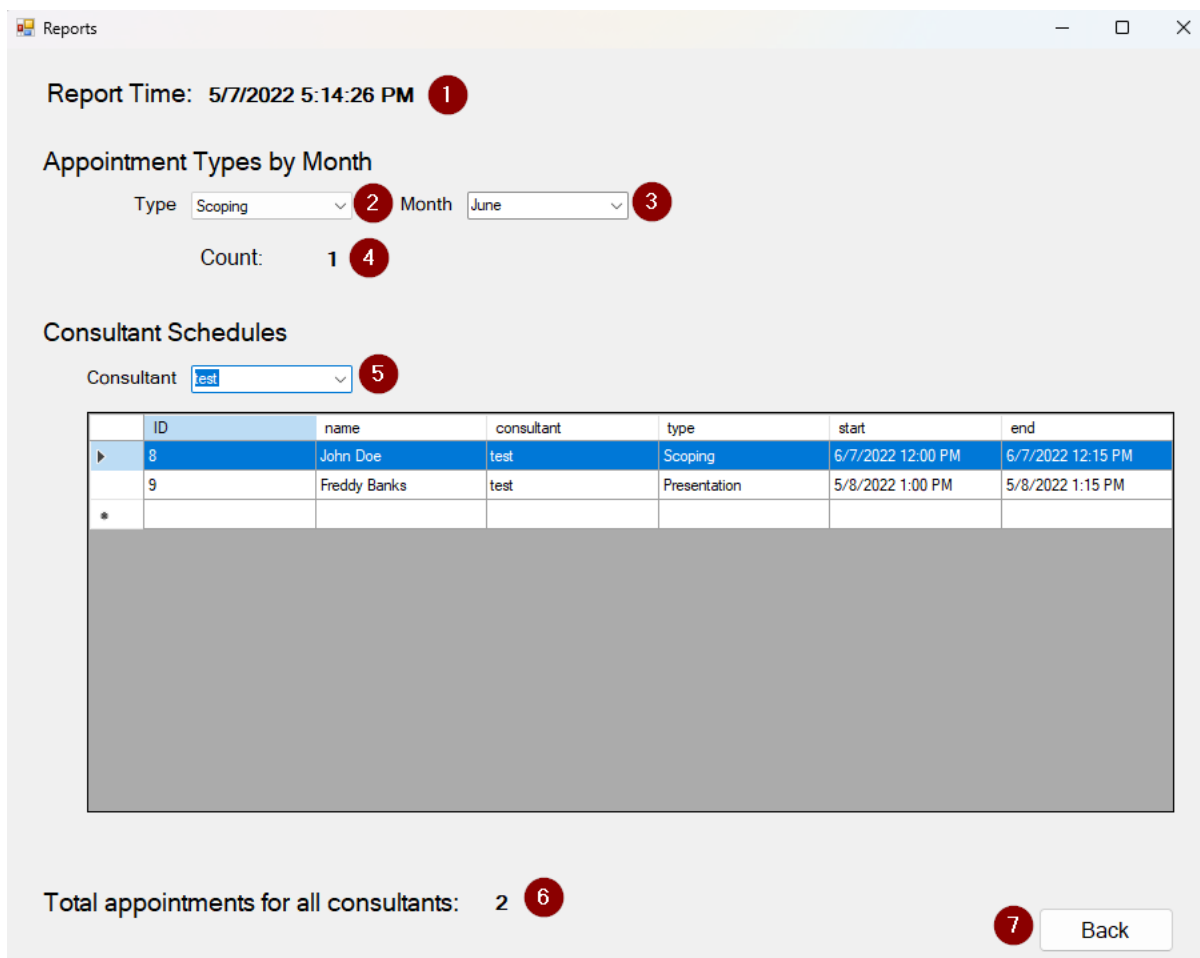
3. Click Reset (3) to return to the original table with all customers.

How To: Use Reports

1. Login to Appointment Keeper.
2. Select Reports on the main screen in the bottom left.



3. Make a note of the time the report is generated (1).
4. For the Appointment Types by Month report, select an Appointment Type from the dropdown (2). Select a month (3). The number of appointments in that month for the selected type will display below (4).
5. To view a specific Consultant Schedule, select the consultant's name from the dropdown (5).
6. View the total number of appointments for all consultants at the bottom (6).
7. To return to the main screen, select the back button (7).



Report Time: 5/7/2022 5:14:26 PM 1

Appointment Types by Month

Type Scoping 2 Month June 3

Count: 1 4

Consultant Schedules

Consultant test 5

	ID	name	consultant	type	start	end
▶	8	John Doe	test	Scoping	6/7/2022 12:00 PM	6/7/2022 12:15 PM
	9	Freddy Banks	test	Presentation	5/8/2022 1:00 PM	5/8/2022 1:15 PM
*						

Total appointments for all consultants: 2 6

7 Back

Administrator Guide

Introduction

The purpose of the Administrator Guide is for Project1 administrators to manage AppointmentKeeper users and review Access Logs for troubleshooting or auditing purposes.

MySQL Workbench setup

1. User Account Administration requires access to MySQL Workbench. Download it [here](#).

Other Downloads:

Windows (x86, 64-bit), MSI Installer	8.0.29	42.9M	Download
(mysql-workbench-community-8.0.29-winx64.msi)		MD5: 2c40a1e64dd8be34c91501749961dd9c Signature	

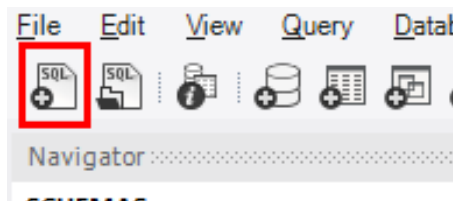
2. After installation, launch MySQL Workbench. Select the plus sign in the center to add a new connection.



3. Enter the following information:
Connection Name: appkeeperdb
Hostname: appkeeperdb.mysql.database.azure.com
Port: 3306
Username: appkeeperadmin@appkeeperdb
Password: WGUcapstone890!!
4. When the connection is created, select the connection. You are now connected to ApplicationKeeper's MySQL Database!

How To: Add new Users

1. Open MySQL Workbench and connect to the database (see steps above for help).
2. In the upper left, select the icon to create a new Query editor.



3. Copy and paste the following MySQL code into the editor:

Project1: AppointmentKeeper Solution

```
SET @username := "username";  
SET @password := "userpassword";
```

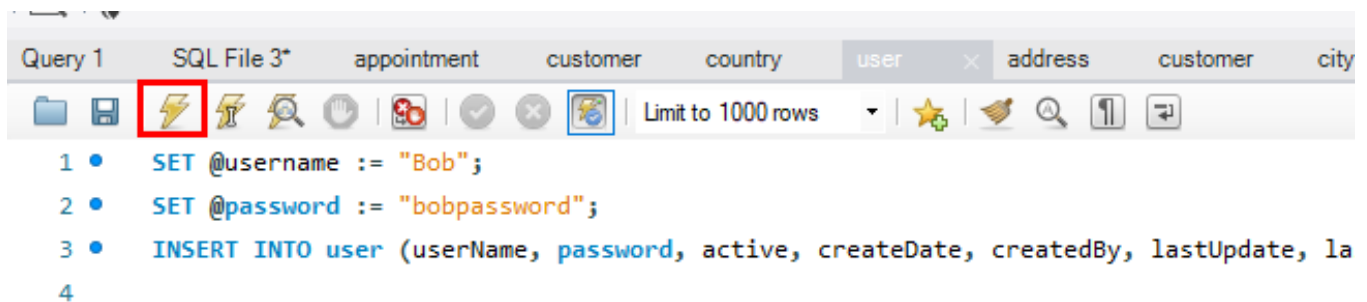
```
INSERT INTO user (userName, password, active, createDate, createdBy, lastUpdate, lastUpdateBy)  
VALUE ("@username", "@password", "1", "2019-01-01 00:00:00", "admin", "2019-01-01 00:00:00",  
"admin")
```

4. On the first line, update the username within quotes to the new user's username. Similarly, update the password. For example, here's how to create a new user with the username "Bob" and password "bobpassword".

```
SET @username := "Bob";  
SET @password := "bobpassword";
```

```
INSERT INTO user (userName, password, active, createDate, createdBy, lastUpdate, lastUpdateBy)  
VALUE (@username, @password, "1", "2019-01-01 00:00:00", "admin", "2019-01-01 00:00:00",  
"admin")
```

5. After editing the query, select the run button in the toolbar:



6. Congratulations! A new user is created. Provide the new username and password to the consultant so they can log in.

How To: Delete Users

1. Follow steps 1 and 2 from "How To: Add new User"
2. Enter the following query in the query editor:

```
SET @username := "username";  
SET @password := "userpassword";
```

```
SELECT userId FROM user WHERE userName = @username AND password = @password;
```

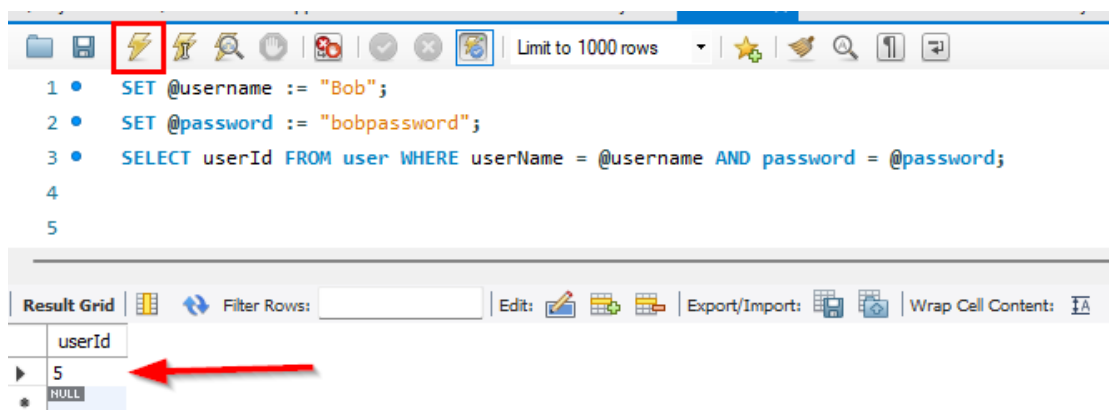
3. On the first line, update the username within quotes to the username and password that needs to be deleted. Similarly, update the password. For example, here's how to delete a user with username "Bob" and password "bobpassword".

Project1: AppointmentKeeper Solution

```
SET @username := "Bob";  
SET @password := "bobpassword";
```

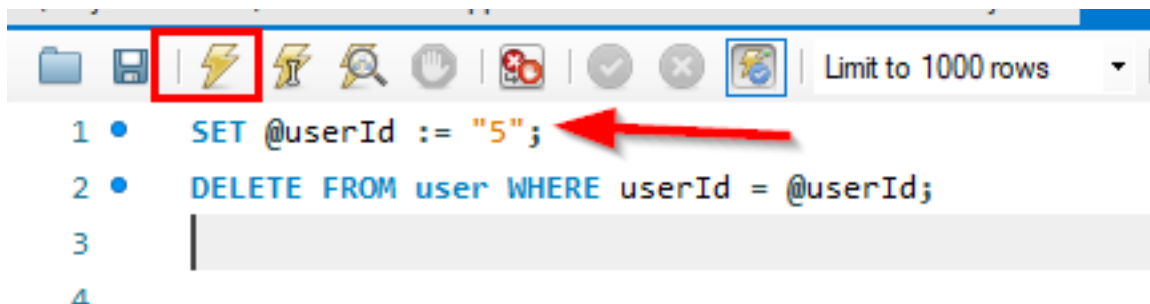
```
INSERT INTO user (userName, password, active, createDate, createdBy, lastUpdate, lastUpdateBy)  
VALUE ("@username", "@password", "1", "2019-01-01 00:00:00", "admin", "2019-01-01 00:00:00",  
"admin")
```

- After editing the query, select the run button in the toolbar. Make a note of the userId in the result window.



- Copy and paste the query below into the query editor. Take the userId retrieved from Step 4 and enter it into the line like in the screenshot below. Execute the query by clicking the lightning bolt icon to delete the user.

```
SET @userId := "5";  
DELETE FROM user WHERE userId = @userId;
```



How To: Review Access Logs

1. ApplicationKeeper includes Access Logs stored in the cloud. When a user login is successful or failed, a new log entry is written to the cloud Access Log.
2. Open this link in your browser to [view the Access Log](#) (internet is required).
3. Here is a sample of Access Logs:

```
5/7/2022 4:08:39 PM - Successful login by test
5/7/2022 4:08:47 PM - Failed login by abc
5/7/2022 4:34:35 PM - Successful login by test
5/7/2022 4:35:49 PM - Successful login by test
5/7/2022 4:40:37 PM - Successful login by test
5/7/2022 4:46:28 PM - Successful login by test
```