

# **STUDENT INFORMATION**

**NAME:** DAISY LOPEZ ADHIAMBO

**REGISTRATION NUMBER:**

# PROJECT TITLE

**EBuilder:**Proposed Resume Builder Desktop  
Application

# ABSTRACT

This document will cover the following areas:

1. Development Environment
2. Functional Requirements
3. System Models
  - 3.1 Use Case Diagram
  - 3.2 Context Diagram
  - 3.3 Level 1 DFD
  - 3.4 User Data Model
  - 3.5 Class Diagram
4. User Interface Design
  - 4.1 Splash Screen
  - 4.2 Home Panel
  - 4.3 Personal Information Section
  - 4.4 Objectives Section
  - 4.5 Skills Section
  - 4.6 Experience Section
  - 4.7 Education Section
  - 4.8 Referees Section
  - 4.9 Save As Section
5. Algorithms: Psuedocodes

# **1.DEVELOPMENT ENVIRONMENT**

**IDE:** IntelliJ IDEA Community Edition

**Programming language:** Java

**GUI Library:** Java Swing

**System Modelling:** StarUML

**User Interface Design:** PENCIL

**Deployment:** Inno Setup

# 2. FUNCTIONAL REQUIREMENTS

**REQ-1:** The app shall allow the job seeker to create a new resume by clicking the “create new resume” button.

**REQ-2:** The app shall allow the job seeker to fill in personal information details.

**REQ-3:** The app shall allow the job seeker to fill in objectives information.

**REQ-4:** The app shall allow the job seeker to fill in the skills information.

**REQ-5:** The app shall allow the job seeker to fill in at least one experience details.

**REQ-6:** The app shall allow the job seeker to fill in primary education details.

**REQ-7:** The app shall allow the job seeker to fill in at least one tertiary education details.

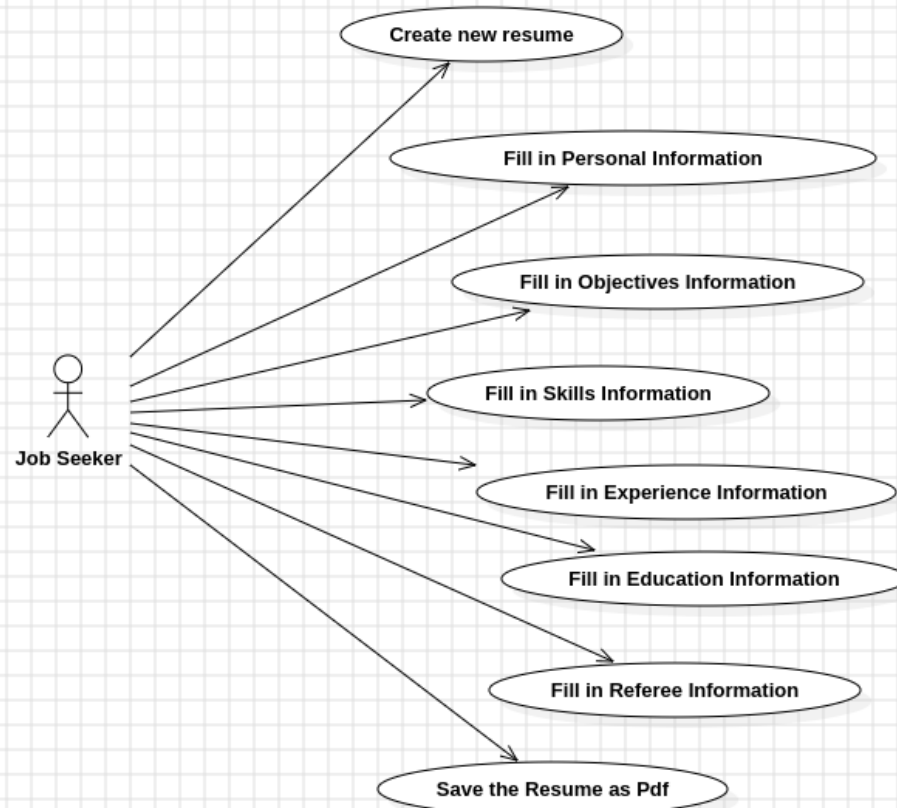
**REQ-8:** The app shall allow the job seeker to fill in information of at least two referees.

**REQ-9:** The app shall allow the job seeker to save the resume as a portable file.

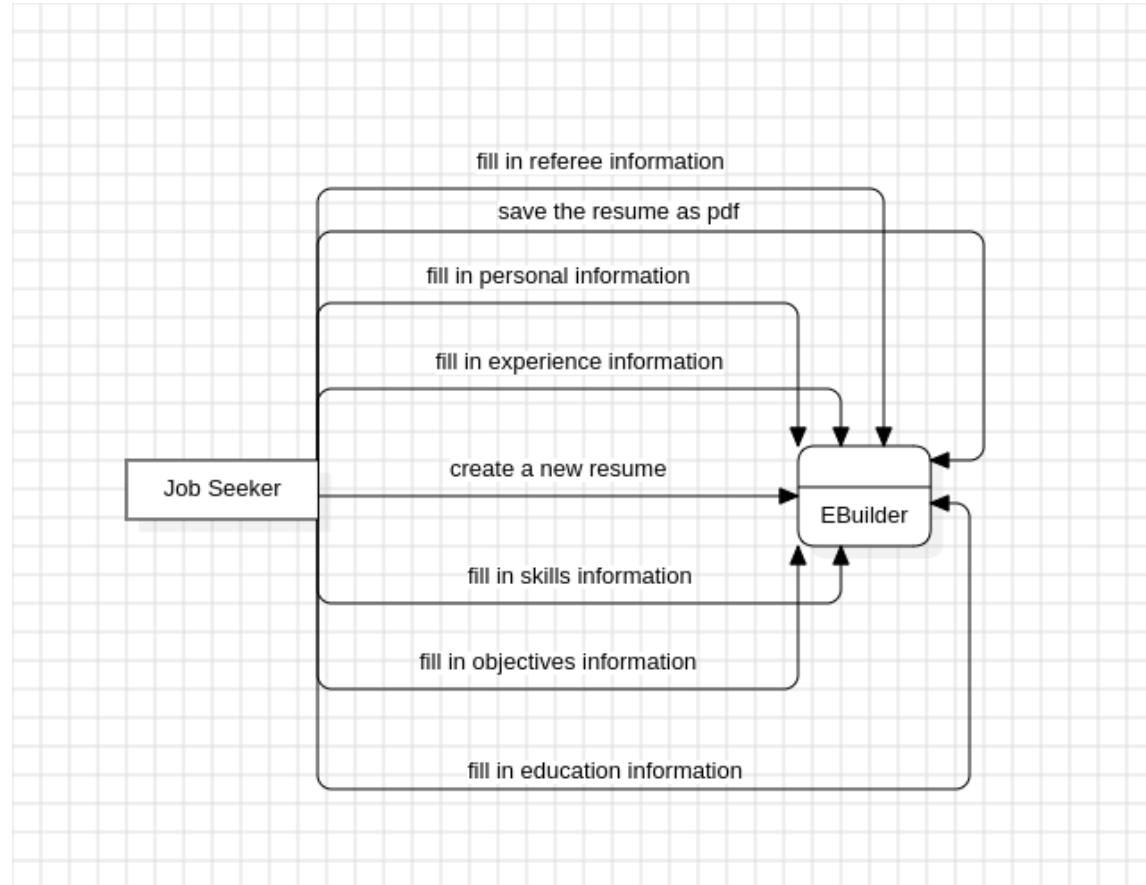
**REQ-10:** The app shall allow the user to navigate from one section to the next by clicking the next button.

# 3.1 Job Seeker

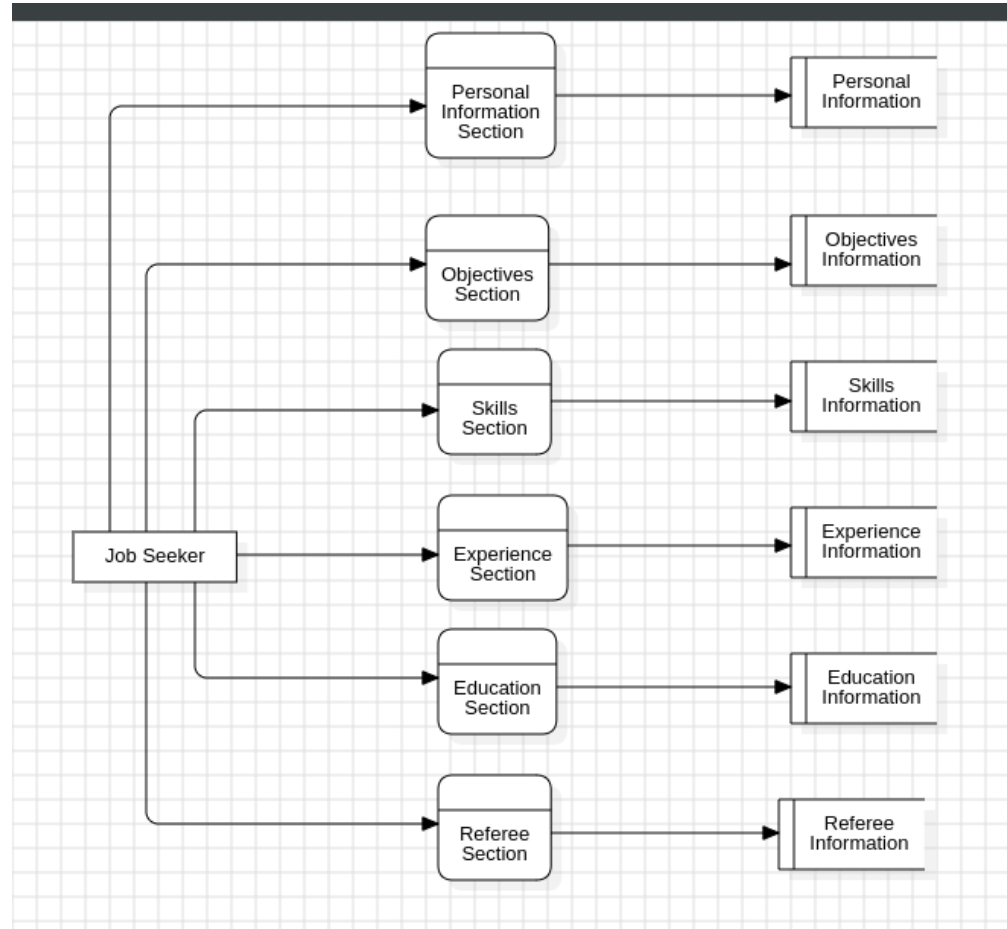
---



## 3.2 CONTEXT DIAGRAM

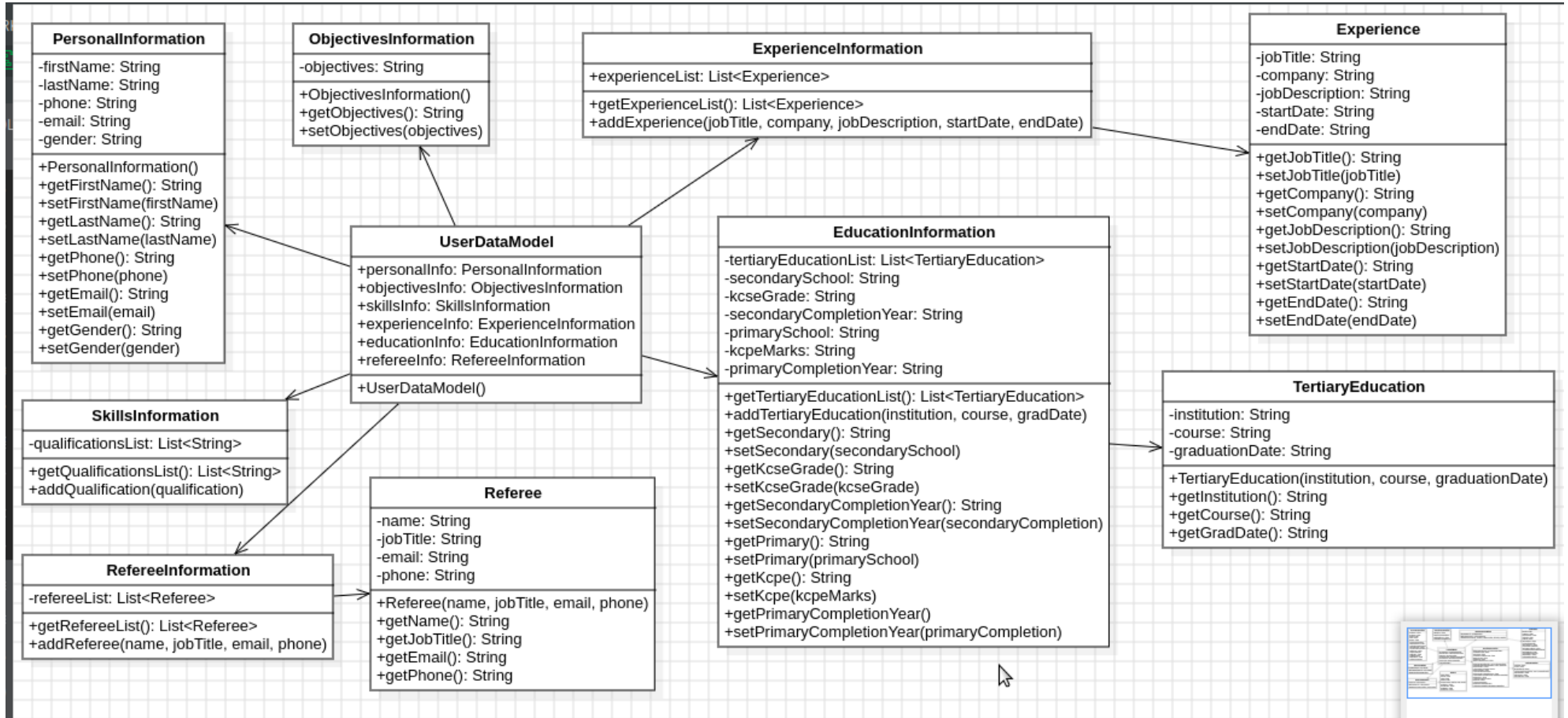


# 3.3 LEVEL 1 DFD

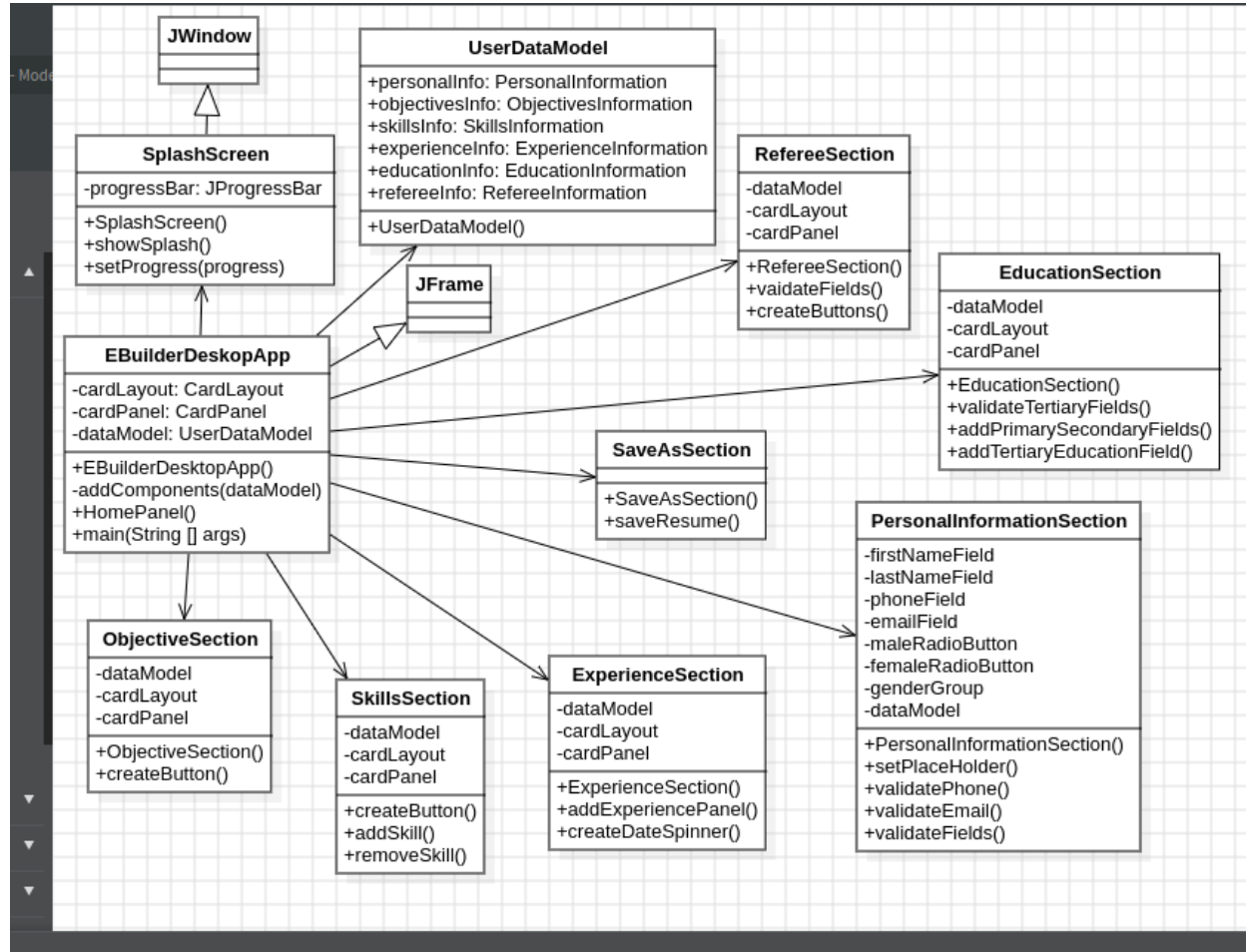




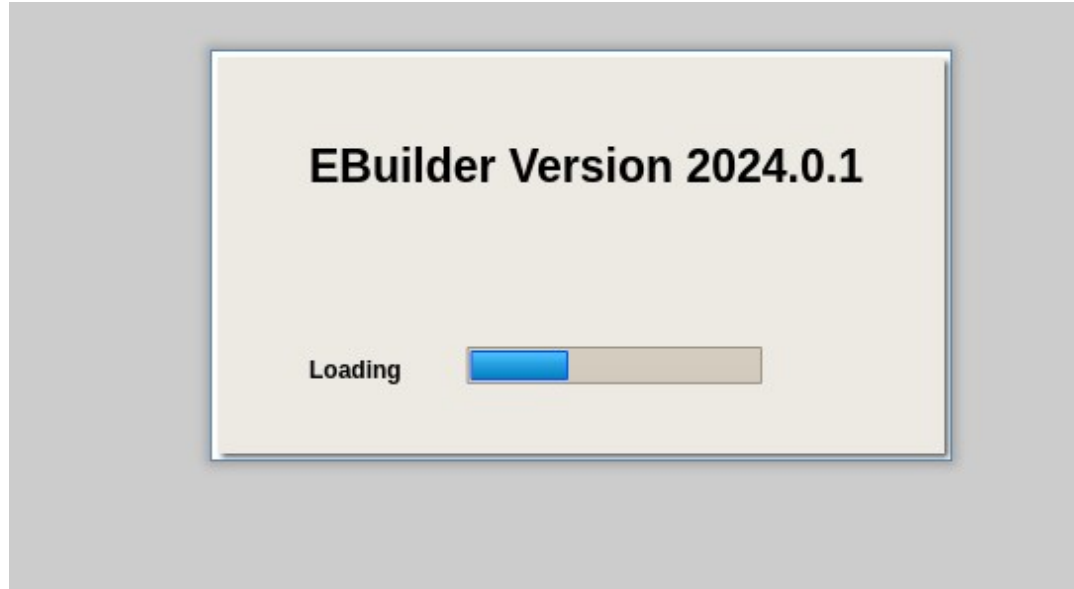
# 3.4 User Data Model



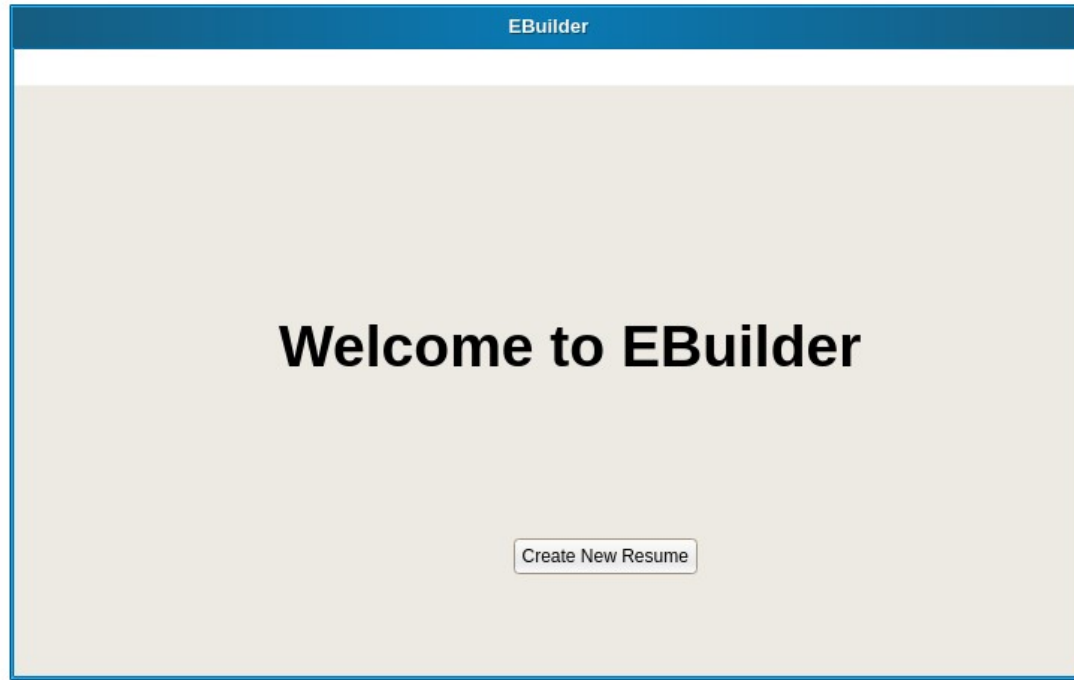
# 3.5 CLASS DIAGRAM



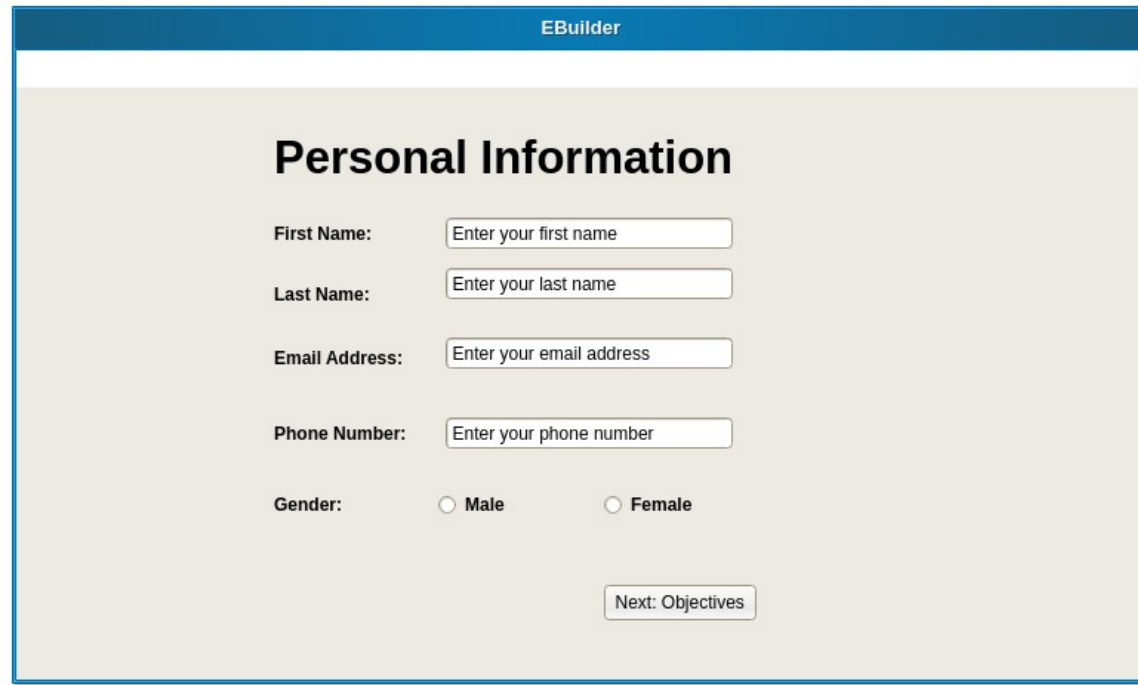
# 4.1 Splash Screen



# 4.2 Home Panel



# 4.3 Personal Information Section



The screenshot shows a web form titled "EBuilder" at the top. Below the title is a section header "Personal Information". The form contains five input fields: "First Name:" with a placeholder "Enter your first name", "Last Name:" with a placeholder "Enter your last name", "Email Address:" with a placeholder "Enter your email address", and "Phone Number:" with a placeholder "Enter your phone number". Below these fields is a "Gender:" label followed by two radio buttons labeled "Male" and "Female". At the bottom right of the form is a button labeled "Next: Objectives".

**EBuilder**

## Personal Information

First Name:

Last Name:

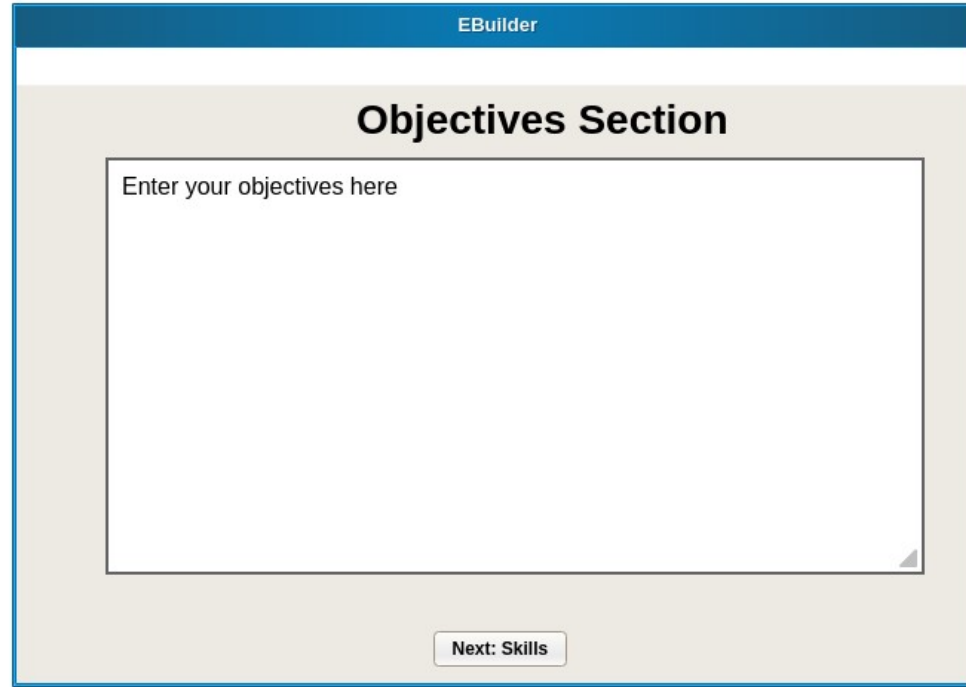
Email Address:

Phone Number:

Gender: ☐ Male ☐ Female

# 4.5 Objectives Section

---



The image shows a screenshot of a web application window titled "EBuilder". Inside the window, there is a section titled "Objectives Section". Below this title is a large text input area with the placeholder text "Enter your objectives here". At the bottom right of the input area, there is a small grey triangle icon. Below the input area, there is a button labeled "Next: Skills".

EBuilder

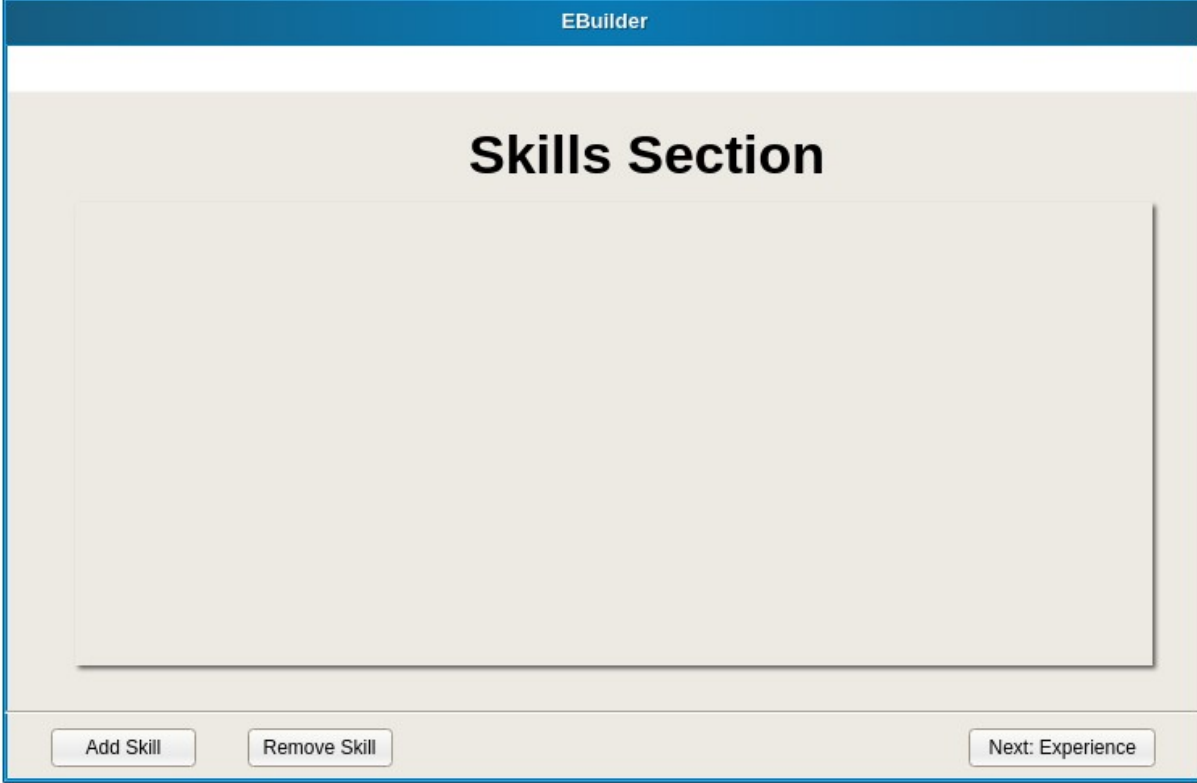
**Objectives Section**

Enter your objectives here

Next: Skills

# 4.6 Skills Section

---



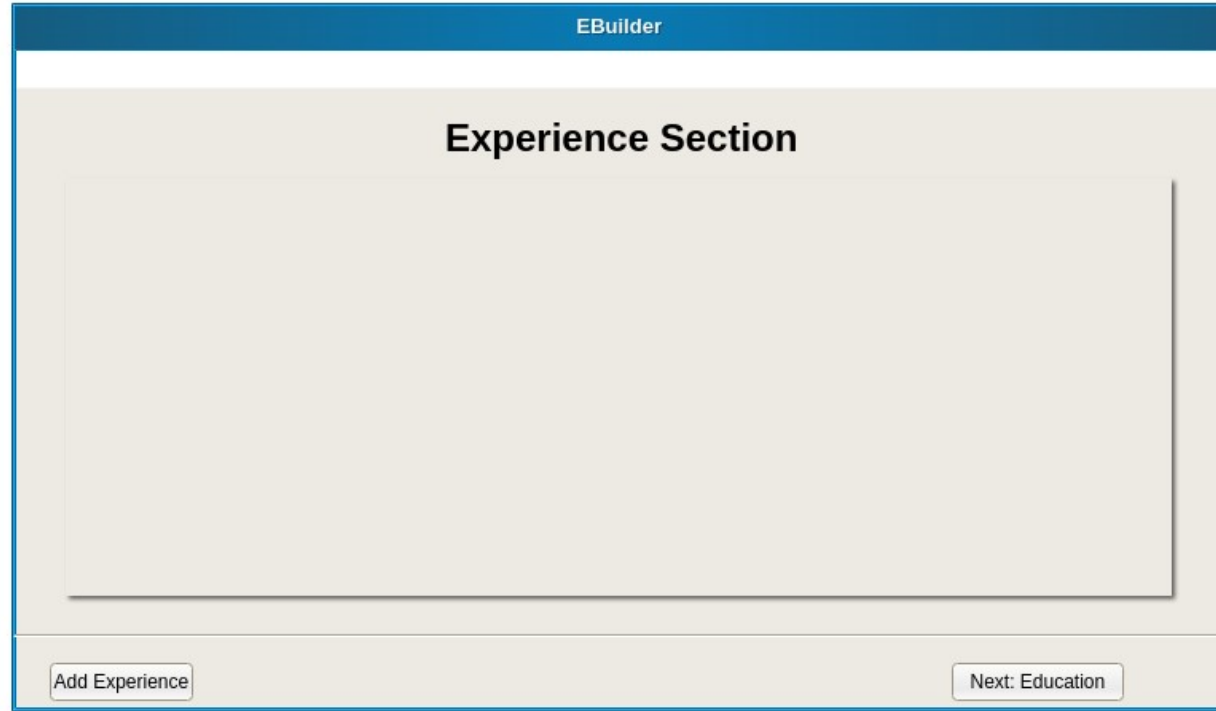
The screenshot shows a web application window titled "EBuilder". Inside the window, the main heading is "Skills Section". Below the heading is a large, empty rectangular box, likely intended for a list of skills. At the bottom of the window, there are three buttons: "Add Skill", "Remove Skill", and "Next: Experience".

EBuilder

## Skills Section

Add Skill Remove Skill Next: Experience

# 4.7 Experience Section



The image shows a software window titled "EBuilder". Inside the window, the text "Experience Section" is centered at the top. Below this text is a large, empty rectangular box with a light beige background and a thin grey border, intended for user input. At the bottom of the window, there are two buttons: "Add Experience" on the left and "Next: Education" on the right.

EBuilder

Experience Section

Add Experience

Next: Education



# 4.8 Education Section

---

**EBuilder**

### Basic Education

**Primary School:**

**KCPE Marks:**

**Primary Completion Year**

**Secondary School:**

**KCSE Grade:**

**Secondary Completion Year:**

### Tertiary Education

**Tertiary Institution:**

**Course:**

**Graduation Date:**

# 4.9 Referees Section

---

EBuilder

Referees Section

Name:

Enter Referee Name

Job Title:

Enter Referee Job Title

Phone Number:

Enter Referee Phone Number

Email:

Enter Referee Email

Name:

Enter Referee Name

Job Title:

Enter Referee Job Title

Phone Number:

Enter Referee Phone Number

Email:

Enter Referee Email

Next: Save As

# 4.10 Save As Section



# 5.1 EBuilderDesktopApp

```
Class EBuilderDesktopApp is a subclass of JFrame
Define private final attributes:
    cardLayout of type CardLayout
    cardPanel of type JPanel
    dataModel of type UserDataModel

Constructor EBuilderDesktopApp
    Call super constructor with title "EBuilder"
    Set default close operation to EXIT_ON_CLOSE
    Set extended state to MAXIMIZED_BOTH
    Instantiate dataModel as a new UserDataModel
    Instantiate cardLayout as a new CardLayout
    Instantiate cardPanel as a new JPanel with cardLayout
    Add cardPanel to content pane with BorderLayout.CENTER
    Call addComponents method with dataModel as argument
    Set the frame visible

Method addComponents with parameter dataModel
    Add homePanel instance to cardPanel with name "homePanel"
    Add PersonalInformationSection instance to cardPanel with name "PersonalInformationSection"
    Add ObjectiveSection instance to cardPanel with name "ObjectiveSection"
    Add HighlightsQualificationsSection instance to cardPanel with name "HighlightsQualificationsSection"
    Add ExperienceSection instance to cardPanel with name "ExperienceSection"
    Add EducationSection instance to cardPanel with name "EducationSection"
    Add ReferencesSection instance to cardPanel with name "ReferencesSection"
    Add SaveAsSection instance to cardPanel with name "SaveAsSection"
    Show "HomePanel" in cardPanel

Inner class HomePanel is a subclass of JPanel
Constructor HomePanel
    Set layout to BorderLayout
    Set background color to (240, 240, 240)
    Create logoLabel as a JLabel with ImageIcon "ebuilder.png"
    Set horizontal alignment of logoLabel to CENTER
    Add logoLabel to BorderLayout.NORTH
    Create welcomeLabel as a JLabel with text "Welcome to EBuilder"
    Set horizontal alignment of welcomeLabel to CENTER
    Set foreground color to (59, 89, 152)
    Set font to Segoe UI, bold, size 24
    Add welcomeLabel to BorderLayout.CENTER
    Create buttonPanel as a JPanel with FlowLayout.CENTER
    Set background color of buttonPanel to (240, 240, 240)
    Create newResumeButton as a JButton with text "Create New Resume"
    Set background color to (59, 89, 152)
    Set foreground color to WHITE
    Set font to Segoe UI, bold, size 18
    Add action listener to newResumeButton to switch cardLayout to "PersonalInformationSection"
    Add newResumeButton to buttonPanel
    Add buttonPanel to BorderLayout.SOUTH

Method main
    Instantiate SplashScreen
    Call showSplash method on SplashScreen
    Try:
        Loop from 0 to 100
            Sleep for 50 milliseconds
        Set progress of SplashScreen
    Catch InterruptedException and print stack trace
    Dispose SplashScreen
    Schedule instantiation of EBuilderDesktopApp on SwingUtilities.invokeLater
```

# 5.2 SplashScreen

Final class SplashScreen is a subclass of JWindow

Define private final attribute:

progressBar of type JProgressBar

Constructor SplashScreen

Instantiate progressBar as a new JProgressBar

Set string painted property of progressBar to true

Instantiate content as a new JPanel with BorderLayout

Create label as a new JLabel with text "EBuilder Loading..."

Set foreground color of label to (59, 89, 152)

Set font of label to Segoe UI, bold, size 30

Set horizontal alignment of label to CENTER

Add label to content with BorderLayout.NORTH

Add progressBar to content with BorderLayout.CENTER

Set content pane of SplashScreen to content

Set size of SplashScreen to (400, 200)

Set location of SplashScreen to be centered

Set SplashScreen to always be on top

Method showSplash

Set SplashScreen visible

Method setProgress with parameter progress

Set value of progressBar to progress

Set string of progressBar to progress + "%"

# 5.3 UserDataModel

Final class UserDataModel

Define public attributes:

- personalInfo of type PersonalInformation
- objectivesInfo of type ObjectivesInformation
- highlightsInfo of type HighlightsOfQualificationsInfo
- experiencesInfo of type ExperienceInformation
- educationInfo of type EducationInformation
- refereeInfo of type RefereeInformation

Constructor UserDataModel

- Instantiate personalInfo as a new PersonalInformation object
- Instantiate objectivesInfo as a new ObjectivesInformation object
- Instantiate highlightsInfo as a new HighlightsOfQualificationsInfo object
- Instantiate experiencesInfo as a new ExperienceInformation object
- Instantiate educationInfo as a new EducationInformation object
- Instantiate refereeInfo as a new RefereeInformation object

# 5.4 PersonalInformation

```
Final class PersonalInformation
Define private attributes:
    firstName of type String
    lastName of type String
    phone of type String
    email of type String
    gender of type String

Constructor PersonalInformation
    // Empty constructor

Method getFirstName
    Return firstName

Method setFirstName with parameter firstName
    Set this.firstName to firstName

Method getLastName
    Return lastName

Method setLastName with parameter lastName
    Set this.lastName to lastName

Method getPhone
    Return phone

Method setPhone with parameter phone
    Set this.phone to phone

Method getEmail
    Return email

Method setEmail with parameter email
    Set this.email to email

Method getGender
    Return gender

Method setGender with parameter gender
    Set this.gender to gender
```

# 5.5 PersonalInformationSection

```
Final class PersonalInformationSection is a subclass of JPanel
Define private final attributes:
    firstNameField of type JTextField
    lastNameField of type JTextField
    phoneField of type JTextField
    emailField of type JTextField
    maleRadioButton of type JRadioButton
    femaleRadioButton of type JRadioButton
    genderGroup of type ButtonGroup
    dataModel of type UserDatabaseModel
Constructor PersonalInformationSection with parameters cardLayout, cardPanel, and dataModel
Assign dataModel to this.dataModel
Set layout to GridBagLayout
Set background color to (245, 245, 245)
Create GridBagConstraints gbc
Set gbc.gridx to 0
Set gbc.gridy to 0
Set gbc.gridwidth to 2
Set gbc.insets to Insets(10, 10, 10, 10)
Set gbc.anchor to WEST
Create headingLabel as a JLabel with text "Personal Information Section"
Set foreground color to (25, 89, 152)
Set text to First, last, size 20
Set horizontal alignment to CENTER
Add headingLabel to this with gbc
Increment gbc.gridx
Set gbc.gridwidth to 1
Create firstNameLabel as a JLabel with text "First Name"
Add firstNameLabel to this with gbc
Increment gbc.gridx
Create lastNameField as a JTextField with size 20
Call setPlaceholder method with firstNameField and "Enter your first name"
Add firstNameField to this with gbc
Repeat the above steps for lastNameLabel, lastNameField, addressLabel, emailField, phoneLabel, and phoneField
Increment gbc.gridx and gbc.gridy
Create genderLabel as a JLabel with text "Gender:"
Add genderLabel to this with gbc
Increment gbc.gridx
Create maleRadioButton and femaleRadioButton as JRadioButton
Create genderGroup as a ButtonGroup
Add maleRadioButton and femaleRadioButton to genderGroup
Create genderPanel as a JPanel
Set panel layout to flow
Add maleRadioButton and femaleRadioButton to genderPanel
Add genderPanel to this with gbc
Increment gbc.gridx and gbc.gridy
Create nextButton as a JButton with text "Next Objectives"
Set background color to (25, 89, 152)
Set foreground color to WHITE
Add nextButton to this with gbc
Add document listener to phoneField and emailField to validate phone and email input
Add action listener to nextButton to validate fields and switch cardLayout to "ObjectiveSection" if validation passes
Method setPlaceholder with parameters textField and placeholder
Set foreground color of textField to GRAY
Set text of textField to placeholder
Add focus listener to textField to handle placeholder behavior
Method validatePhone
Define phoneRegex as "[0-9]{10}"
Create matcher from phoneRegex
Create matcher from phoneField text
Set foreground color of phoneField red based on whether it matches phoneRegex
Method validateEmail
Define emailRegex as "[a-zA-Z0-9]{1,30}@[a-zA-Z0-9]{1,30}.[a-zA-Z]{2,4}"
Create matcher from emailRegex
Create matcher from emailField text
Set foreground color of emailField based on whether it matches emailRegex
Method validateFields
Set the name, last name, email, and phone in dataModel.personality
Set gender in dataModel.personality based on selected radioButton
Return true if all mandatory fields are filled and phone/email validation passes, otherwise false
```



# 5.6 ObjectivesInformation

Final class ObjectivesInformation

Define private attribute:

objectives of type String

Method getObjectives

Return objectives

Method setObjectives with parameter objectives

Set this.objectives to objectives

# 5.7 ObjectivesSection

```
Final class ObjectivesSection is a subclass of JPanel
Define private final attributes:
    dataModel of type UserDataModel
    cardLayout of type CardLayout
    cardPanel of type JPanel

Constructor ObjectivesSection with parameters cardLayout, cardPanel, and dataModel
Set cardLayout to provided cardLayout
Set cardPanel to provided cardPanel
Set dataModel to provided dataModel
Set layout to BorderLayout
Set background color to WHITE

Create sectionLabel as a new JLabel with text "Objectives Section"
Set font of sectionLabel to Segoe UI, bold, size 20
Set foreground color of sectionLabel to (59, 89, 152)
Set horizontal alignment of sectionLabel to CENTER
Add sectionLabel to BorderLayout.NORTH

Create objectiveTextPane as a new JTextPane
Set font of objectiveTextPane to Segoe UI, plain, size 16
Set background color of objectiveTextPane to WHITE
Set foreground color of objectiveTextPane to (74, 74, 74)
Create scrollPane as a new JScrollPane with objectiveTextPane
Set border of scrollPane to empty border with insets (10, 10, 10, 10)
Add scrollPane to BorderLayout.CENTER

Create tooltipLabel as a new JLabel
Set text of tooltipLabel to "<html><p style='width: 200px;'>Enter your career objectives here.</p></html>"
Set tool tip text of objectiveTextPane to "Enter your career objectives here."
Add mouse motion listener to objectiveTextPane:
    When mouse is moved:
        Get mouse point
        Try:
            Convert mouse point to view coordinates
            If converted rectangle is not null:
                Create rectangle from converted coordinates
                Set bounds of tooltipLabel to position below the rectangle
                Make tooltipLabel visible
            Catch IllegalArgumentException and hide tooltipLabel
Add tooltipLabel to BorderLayout.SOUTH

Create buttonPanel as a new JPanel with FlowLayout.CENTER
Create nextButton as a JButton with text "Next: Skills" and background color (59, 89, 152)
Set foreground color of nextButton to WHITE
Add nextButton to buttonPanel
Add buttonPanel to BorderLayout.SOUTH

Add action listener to nextButton:
    When clicked:
        Switch cardLayout to "HighlightsQualificationsSection" in cardPanel
        Set objectives in dataModel to content of objectiveTextPane

Method createButton with parameters text and background
Create button as a new JButton with provided text and background color
Set foreground color of button to WHITE
Return button
```

# 5.8 SkillsInformation

Final class HighlightsOfQualificationsInfo

Define private final attribute:

qualificationsList of type List<String> initialized to an empty ArrayList<String>

Method getQualificationsList

Return qualificationsList

Method addQualification with parameter qualification

Add qualification to qualificationsList

Method removeQualification with parameter qualification

Remove qualification from qualificationsList

# 5.9 SkillsInformationSection

Class HighlightsCQualificationsSection is a subclass of JPanel

Define private final attributes:  
dataModel of type DataQualifications  
qualPanel of type CardLayout  
cardPanel of type JPanel

Constructor HighlightsCQualificationsSection with parameters cardLayout, cardPanel, and dataModel

Set cardLayout to provided cardLayout  
Set cardPanel to provided cardPanel  
Set dataModel to provided dataModel  
Set layout to BorderLayout  
Set background color to WHITE

Create qualificationsModel as a new DefaultListModel<String>  
Create qualificationsList as a new JList<String> with qualificationsModel  
Create titleBorder as a JTitleBorder with title "Skills Section"  
Set title color of titleBorder to (59, 89, 152)  
Set title font of titleBorder to bold, size 15, and center justification  
Create qualificationsScrollPane as a new JScrollPane with qualificationsList and titleBorder  
Set border of qualificationsScrollPane to BorderLayout  
Add qualificationsScrollPane to BorderLayout.CENTER

Create addButton as a JButton with text "Add Skill" and background color (59, 89, 152)  
Create removeButton as a JButton with text "Remove Skill" and background color (227, 28, 36)

Set background color of qualificationsList to (240, 240, 240)  
Set selection background color of qualificationsList to (173, 225, 47)

Add action listener to addButton:  
When clicked:  
Call addQualification method with qualificationsModel

Add action listener to removeButton:  
When clicked:  
Call removeQualification method with qualificationsList and qualificationsModel

Create buttonPanelLeft as a JPanel with FlowLayout LEFT  
Add addButton and removeButton to buttonPanelLeft

Create buttonPanelRight as a JPanel with FlowLayout RIGHT  
Create navigation as a JButton with text "Next Experience" and background color (59, 89, 152)  
Add navigation to buttonPanelRight

Create navigationButtonPanel as a JPanel with BorderLayout  
Add buttonPanelLeft to WEST and buttonPanelRight to EAST of navigationButtonPanel  
Set empty border to navigationButtonPanel  
Add navigationButtonPanel to BorderLayout.SOUTH

Add action listener to nextButton:  
When clicked:  
Switch cardLayout to "ExperienceSection" in cardPanel

Method createButton with parameters text and background  
Create button as a new JButton with provided text and background color  
Set foreground color of button to WHITE  
Return button

Method addQualification with parameter qualificationsModel  
Display input dialog to get new qualification  
If new qualification is not empty:  
Add new qualification to qualificationsModel  
Add new qualification to dataModel highlightsInfo

Method removeQualification with parameters qualificationsList and qualificationsModel

Get selected index from qualificationsList  
Calculate index to set to 0  
Get removed qualification from qualificationsModel at selected index  
Remove qualification from qualificationsModel  
Remove qualification from dataModel highlightsInfo  
Else:  
Show message dialog indicating no selection

# 5.11 ExperienceInformation

Final class ExperienceInformation

Define public final attribute:

experienceList of type List<Experience> initialized to an empty ArrayList<Experience>

Method getExperienceList

Return experienceList

Method addExperience with parameters jobTitle, company, jobDescription, startDate, endDate

Add new Experience instance to experienceList with provided parameters

Final class Experience

Define private final attributes:

jobTitle of type String

company of type String

jobDescription of type String

startDate of type String

endDate of type String

Constructor Experience with parameters jobTitle, company, jobDescription, startDate, endDate

Set jobTitle to provided jobTitle

Set company to provided company

Set jobDescription to provided jobDescription

Set startDate to provided startDate

Set endDate to provided endDate

Method getJobTitle

Return jobTitle

Method getCompany

Return company

Method getJobDescription

Return jobDescription

Method getStartDate

Return startDate

Method getEndDate

Return endDate

Method toString

Return formatted string representation of experience details

## 5.12 ExperienceSection

[illegible]

# 5.13 EducationInformation

```
Final class EducationInformation
Define private final attributes:
    tertiaryEducationList of type List<TertiaryEducation> initialized to an empty ArrayList
    secondarySchool of type String
    kcaeGrade of type String
    secondaryCompletionYear of type String
    primarySchool of type String
    kcpaMarks of type String
    primaryCompletionYear of type String

Method getTertiaryEducationList() List<TertiaryEducation>
    Return tertiaryEducationList

Method addTertiaryEducation with parameters institution, course, and graduationDate
    Add a new TertiaryEducation object with provided details to tertiaryEducationList

Method getSecondarySchool() String
    Return secondarySchool

Method setSecondarySchool with parameter secondarySchool
    Set the value of secondarySchool attribute to the provided value

Method getKcaeGrade() String
    Return kcaeGrade

Method setKcaeGrade with parameter kcaeGrade
    Set the value of kcaeGrade attribute to the provided value

Method getSecondaryCompletionYear() String
    Return secondaryCompletionYear

Method setSecondaryCompletionYear with parameter secondaryCompletionYear
    Set the value of secondaryCompletionYear attribute to the provided value

Method getPrimarySchool() String
    Return primarySchool

Method setPrimarySchool with parameter primarySchool
    Set the value of primarySchool attribute to the provided value

Method getKcpaMarks() String
    Return kcpaMarks

Method setKcpaMarks with parameter kcpaMarks
    Set the value of kcpaMarks attribute to the provided value

Method getPrimaryCompletionYear() String
    Return primaryCompletionYear

Method setPrimaryCompletionYear with parameter primaryCompletionYear
    Set the value of primaryCompletionYear attribute to the provided value

Final class TertiaryEducation
Define private final attributes:
    institution of type String
    course of type String
    graduationDate of type String

Constructor TertiaryEducation with parameters institution, course, and graduationDate
    Set the value of institution, course, and graduationDate attributes to the provided values

Method getInstitution() String
    Return institution

Method getCourse() String
    Return course

Method getGraduationDate() String
    Return graduationDate

Method toString() String
    Return a string representation containing institution, course, and graduationDate details
```

## 5.14 EducationSection

[illegible]



# 5.16 RefereeInformation

Final class RefereeInformation

Define private attribute refereeList as a List of Referee objects

Method getRefereeList

Return refereeList

Method addReferee with parameters name, jobTitle, email, and phone

Create a new Referee object with provided parameters

Add the new Referee object to refereeList

Final class Referee

Define private attributes:

name of type String

jobTitle of type String

email of type String

phone of type String

Constructor Referee with parameters name, jobTitle, email, and phone

Set name, jobTitle, email, and phone attributes to provided values

Method getName

Return name

Method getJobTitle

Return jobTitle

Method getEmail

Return email

Method getPhone

Return phone

Method toString

Return a string representation of Referee with name, jobTitle, email, and phone attributes

## 5.17 RefereeSection

1. **Systemanforderungen** (z.B. Betriebssystem, Hardware, Netzwerk)  
 2. **Benutzeranforderungen** (z.B. Rollen, Berechtigungen, Schnittstellen)  
 3. **Funktionsanforderungen** (z.B. Kernfunktionen, Erweiterungen)  
 4. **Qualitätsanforderungen** (z.B. Performance, Sicherheit, Zuverlässigkeit)  
 5. **Integration** (z.B. Schnittstellen, Datenbanken)  
 6. **Testanforderungen** (z.B. Testfälle, Testumgebung)  
 7. **Dokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 8. **Implementierung** (z.B. Programmierung, Konfiguration)  
 9. **Deployment** (z.B. Installation, Migration)  
 10. **Wartung** (z.B. Updates, Fehlerbehebungen)  
 11. **Entsorgung** (z.B. Archivierung, Löschung)  
 12. **Compliance** (z.B. Datenschutz, Sicherheitsstandards)  
 13. **Benutzerakzeptanz** (z.B. Schulung, Feedback)  
 14. **Projektmanagement** (z.B. Zeitplan, Ressourcen)  
 15. **Risikoprüfung** (z.B. Identifizierung, Bewertung)  
 16. **Veränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 17. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 18. **Systemwartung** (z.B. Monitoring, Support)  
 19. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 20. **Systemarchitektur** (z.B. Design, Implementierung)  
 21. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 22. **Systemtest** (z.B. Testfälle, Testumgebung)  
 23. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 24. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 25. **Systemdeployment** (z.B. Installation, Migration)  
 26. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 27. **Systementsorgung** (z.B. Archivierung, Löschung)  
 28. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 29. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 30. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 31. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 32. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 33. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 34. **Systemwartung** (z.B. Monitoring, Support)  
 35. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 36. **Systemarchitektur** (z.B. Design, Implementierung)  
 37. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 38. **Systemtest** (z.B. Testfälle, Testumgebung)  
 39. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 40. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 41. **Systemdeployment** (z.B. Installation, Migration)  
 42. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 43. **Systementsorgung** (z.B. Archivierung, Löschung)  
 44. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 45. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 46. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 47. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 48. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 49. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 50. **Systemwartung** (z.B. Monitoring, Support)  
 51. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 52. **Systemarchitektur** (z.B. Design, Implementierung)  
 53. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 54. **Systemtest** (z.B. Testfälle, Testumgebung)  
 55. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 56. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 57. **Systemdeployment** (z.B. Installation, Migration)  
 58. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 59. **Systementsorgung** (z.B. Archivierung, Löschung)  
 60. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 61. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 62. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 63. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 64. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 65. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 66. **Systemwartung** (z.B. Monitoring, Support)  
 67. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 68. **Systemarchitektur** (z.B. Design, Implementierung)  
 69. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 70. **Systemtest** (z.B. Testfälle, Testumgebung)  
 71. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 72. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 73. **Systemdeployment** (z.B. Installation, Migration)  
 74. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 75. **Systementsorgung** (z.B. Archivierung, Löschung)  
 76. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 77. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 78. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 79. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 80. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 81. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 82. **Systemwartung** (z.B. Monitoring, Support)  
 83. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 84. **Systemarchitektur** (z.B. Design, Implementierung)  
 85. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 86. **Systemtest** (z.B. Testfälle, Testumgebung)  
 87. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 88. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 89. **Systemdeployment** (z.B. Installation, Migration)  
 90. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 91. **Systementsorgung** (z.B. Archivierung, Löschung)  
 92. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 93. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 94. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 95. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 96. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 97. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 98. **Systemwartung** (z.B. Monitoring, Support)  
 99. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 100. **Systemarchitektur** (z.B. Design, Implementierung)  
 101. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 102. **Systemtest** (z.B. Testfälle, Testumgebung)  
 103. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 104. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 105. **Systemdeployment** (z.B. Installation, Migration)  
 106. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 107. **Systementsorgung** (z.B. Archivierung, Löschung)  
 108. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 109. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 110. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 111. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 112. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 113. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 114. **Systemwartung** (z.B. Monitoring, Support)  
 115. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 116. **Systemarchitektur** (z.B. Design, Implementierung)  
 117. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 118. **Systemtest** (z.B. Testfälle, Testumgebung)  
 119. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 120. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 121. **Systemdeployment** (z.B. Installation, Migration)  
 122. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 123. **Systementsorgung** (z.B. Archivierung, Löschung)  
 124. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 125. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 126. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 127. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 128. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 129. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 130. **Systemwartung** (z.B. Monitoring, Support)  
 131. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 132. **Systemarchitektur** (z.B. Design, Implementierung)  
 133. **Systemintegration** (z.B. Schnittstellen, Datenbanken)  
 134. **Systemtest** (z.B. Testfälle, Testumgebung)  
 135. **Systemdokumentation** (z.B. Benutzerhandbuch, Wartungsanleitung)  
 136. **Systemimplementierung** (z.B. Programmierung, Konfiguration)  
 137. **Systemdeployment** (z.B. Installation, Migration)  
 138. **Systemwartung** (z.B. Updates, Fehlerbehebungen)  
 139. **Systementsorgung** (z.B. Archivierung, Löschung)  
 140. **Systemcompliance** (z.B. Datenschutz, Sicherheitsstandards)  
 141. **Systembenutzerakzeptanz** (z.B. Schulung, Feedback)  
 142. **Systemprojektmanagement** (z.B. Zeitplan, Ressourcen)  
 143. **Systemrisikoprüfung** (z.B. Identifizierung, Bewertung)  
 144. **Systemveränderungsmanagement** (z.B. Änderungsanträge, Freigabe)  
 145. **Systemüberprüfung** (z.B. Audits, Zertifizierung)  
 146. **Systemwartung** (z.B. Monitoring, Support)  
 147. **Systemevolution** (z.B. Erweiterungen, Upgrades)  
 148.

# 5.18 SaveAsSection

```
First class SaveAsSection extends JPanel
Constructor SaveAsSection with parameters cardLayout, cardPanel, and dataModel
Set layout to BorderLayout
Set background color

Create sectionLabel as JLabel with text "Save As Section", font, color, and alignment
Add sectionLabel to the NORTH of the panel

Create saveButton as JButton with text "Save as PDF", background color, foreground color, font, size, and ActionListener
When clicked:
    Call saveResume method with dataModel as parameter

Create buttonPanel as JPanel with FlowLayout and background color
Add saveButton to buttonPanel
Add buttonPanel to the CENTER of the panel

Method saveResume with parameter dataModel
Try:
    Create fileName as string concatenating last name from dataModel with ".txt"
    Create file with fileName
    Create writer as FileWriter for the file

    Write "RESUME\r\n" to the file

    Format current date as "ddMMyyyy"
    Write "Name: " + first name + last name + " Email: " + email + " Phone: " + phone + " Date: " + formatted date + "\r\n" to the file

    Write "Objectives\r\n" + objectives from dataModel + "\r\n" to the file
    Write "Skills\r\n" to the file
    Iterate over qualificationsList from dataModel:
        Write each skill followed by a newline to the file

    Write "Experience\r\n" to the file
    Iterate over experienceList from dataModel:
        Write each experience followed by a newline to the file

    Write "Education\r\n" to the file
    Iterate over tertiaryEducationList from dataModel:
        Write each tertiary education followed by a newline to the file

    Write "Secondary:" + secondary school name + "\r\n" to the file
    Write "KCSE Grade:" + KCSE grade + "\r\n" to the file
    Write "Year of Completion:" + secondary completion year + "\r\n" to the file

    Write "Primary:" + primary school name + "\r\n" to the file
    Write "KCPE Marks:" + KCPE marks + "\r\n" to the file
    Write "Year of Completion:" + primary completion year + "\r\n" to the file

    Write "References:" to the file
    Iterate over referenceList from dataModel:
        Write each reference followed by a newline to the file

    Close the writer
    Show success message dialog with file name
Catch IOException:
    Print stack trace
    Show error message dialog "Failed to save resume."
```