UNIVERSITÉ Concordia UNIVERSITY

SOEN-6751 HUMAN COMPUTER INTERFACE DESIGN

Instructor Dr. Rajagopalan Jayakumar

Assignment 2

Team members

Sr. No.	Name	Student ID	Email-id
1	Yash Golwala	40085663	golwalayash@gmail.com
2	Chirag Vora	40091454	chirag123192@gmail.com
3	Pooja Dhir	40104545	poojadhir1996@gmail.com
4	Harsh Divecha	40084737	harsh2826@gmail.com
5	Sanjana Udar	40094029	sanjana.udar@gmail.com
6	Jaswanth Banavathu	40080737	jaswanthbanavathu@gmail.com

Table of Contents

Table of Contents

Conceptual Design(s)

Conceptual Design 1

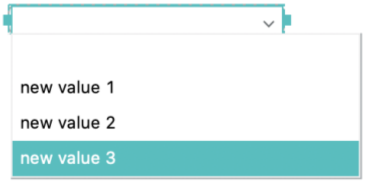
a. Transforming the requirements into conceptual design

How a person rates their capabilities and their level of expertise is relative. So, to avoid confusion for the user and make it more straight forward, the application would ask the user about the number of years of experience that he/she has using GCC and according to that direct the user to the correct interface that would best suit their needs.

Suppose, if the user selects 0-1 years, he/she would be considered a Novice user who has none to minimal knowledge about GCC and would require more assistance with navigation and commands.

If the user selects 3+ years, we can provide the user with the Expert Developer interface which would have less assistance as it would just bore the user but a set of more complex commands which are absent in Novice and Typical user type interface.

b. Selecting Interface Metaphor

Metaphors	Description
❖ Radio Button ◉ ○	It creates an impact on the user's mind to select any one of the available options.
❖ Dropdown list  new value 1 new value 2 new value 3	It depicts the affordance to have a list of commands on any page.
❖ Fill color	As a feedback we will fill the selected option i.e. selected radio button will be marked when user selected that corresponding option.
❖ Windowpane	It will depict the information or the output after the user will interact with the commands option.

❖ Navigation Buttons ⌂ ←	To navigate through the pages at any given time.
❖ Texts (Psychological)	Text will be chosen in a way that will help users to create a good mental model while using the interface.

c. Selecting Interaction Type.

"**Instructing Interaction Type**" will be suitable for this design as it will allow users to interact with the interface without much efforts and can instruct the interface using a mouse or keyboard for achieving the task completion.

"**Conversing Interaction Type**" will also be integrate along with instructing type for making user friendly UI and have good UX. Text will be selected appropriately based on the selected user type.

Conceptual Design 2

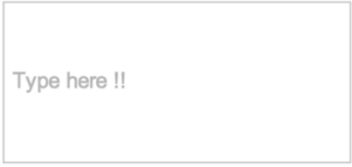
a. Transforming the requirements into conceptual design

The user gets to select from the 3 user types after the application is opened for the 1st time. Suppose the user selects the option of Novice user type at first, the next time he/she starts the application they will be provided with the Novice interface only. If they come to a conclusion that they would like to try or that they're capable enough to start working as a Typical programmer or Expert Developer on the GCC interface they can go to the menu and change the user type to Typical programmer which will provide them with the respective interface and command list.

The menu in the interface would be a drop-down list in the top bar or the side bar, which on being clicked will show the unselected options for the user type. On every interface type this menu option would give users the flexibility of switching from one interface type to another as and when needed.

b. Selecting Interface Metaphor

Metaphors	**Description**
❖ TextArea	Gives idea of writing command or code in the dedicated text area.

 Type here !!	
❖ Buttons Continue	For instructing the Interface to do action. Button when pressed will be highlighted using shadow and green color to give user a feedback for that button being pressed.
❖ Options	This will depict the idea of choosing one out of all of the options.

c. Selecting Interaction Type

"**Instructing User Type**" will be followed while developing this UI. User can instruct using buttons and options from the screen and can write their own code or commands and can execute it by instructing the UI.

Conceptual Design 3

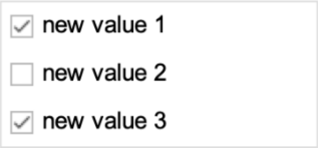
a. Transforming the requirements into conceptual design

Every time the user starts the application the user gets 3 user type options to select form. This is a simple design where the user gets to choose which interface, they would be comfortable using at every instance of starting the application.

The user would have 3 radio buttons or 3 drop down menu options of the user type that they can select from on every instance of the application starting. Once the user is using a particular interface and wants to change it he/she can restart the application and select the same or different user type for the interface that they want to use.

b. Selecting Interface Metaphor

Metaphors	**Description**
❖ Check Box	

 ☑ new value 1 □ new value 2 ☑ new value 3	Users can get feedback of the ticked options (green color ticks).
❖ Menu Bar	Shows the list of menu options in the toolbar on the top of the screen with the quit app button.
❖ Button with shadows	Indicates the future action based on the button selected.
❖ Logo Icons with Picture ▶	Can depict the idea of the related actions. i.e. Terminate Command, Close button, Run Command.

c. Selecting Interaction Type

“**Instructing & Conversing Interaction Type**” can be appropriate for making above metaphors and design work. It will create good mental model in user’s mind whenever user will interact with the interface. He can get idea easily with the logos and picture depicting the set of future actions giving user a good use experience.

Prototyping

What is a Prototype

A prototype is an expression of design intent and it allows designers to present their idea designs and see them in action. It’s a simulation of the final interaction between user and the interface. The main purpose of prototyping is to ensure that the design concept works as intended and also to determine whether the people are able to use a product.

Low-Fidelity Prototype

Fidelity conveys the look and the feel of the final product i.e. the level of details and its realism. Lo-Fi prototyping is quick and easy way to translate high-level design concepts into tangible and testable artifacts. The very first role of lo-fi is test functionality of the product.

Basic Characteristics of Lo-Fi

- **Visual Design:** Include only the essential visual attributes of the final product.
- **Content:** Only key elements of the contents are to be included.
- **Interactivity:** During a test session, a particular person familiar with the design acts as a computer and manually changes the design states in real-time. This can be achieved using wireframes.

Pros:

- **Inexpensive:** Can be developed with extremely low cost.
- **Fast:** Can be created within 10-15 minutes.
- **Collaborative:** Simulates group work and also does not require any special skills, so more people can be included for getting multiple perspective.
- **Clarifying:** Team members and stakeholders will get a very clear idea and expectations about the product which can help to prevent bad UX in future.

Cons:

- **Uncertainty during testing:** Test participants might be unclear about what is supposed to work and isn't. A lot of imagination is required.
- **Limited Interactivity:** Impossible to show the complex animations or transitions using Lo-Fi prototype.

Prototype 1

Story Board (based on scenario)

1. Usability of a Design

a) Usability Goals:

- **Learnability:** Mental model is used to achieve this type of goal. It depends on how user learns new things.
- **Effectiveness:** The interface itself should guide the user to perform the required operations by having appropriate widgets with labels.
- **Utility:** Provides the necessary features which makes the user's tasks easy or fun to use.
- **Efficiency:** Makes the interface more intuitive and should help the user perform the tasks faster.
- **Memorability:** It is easier for users to remember how to carry out a task due to the mental model.

b) Usability Principles:

- **Consistency:** The app is consistent in terms of UI template throughout the application.

- **Visibility:** UI elements and icons which makes its functionality visible/obvious to the user. Ex.: For an error, there should be a popup with the error and a red stop sign.
- **Constraints:** Limits the functionalities according to the level of user or user type.
- **Affordance:** Usage of appropriate UI elements which denotes how to use it to perform the functionality.

2. Error Tolerance of the design

When user enters incorrect combination of GCC command and linking options, the user is prompted with error message and a stack trace on the console.

3. Efficiency of the design

The UI is made simple by consolidating numerous options by categorising them and using appropriate widgets/ UI elements. The simplicity of the design helps provide a mental model to the user which makes the usage of software more efficient.

STORY SCENARIO

Nick Jones is new to programming and curious to learn Smart GCC. He decided to learn Smart GCC, so he takes his laptop and opens Smart GCC software. Firstly, he has to answer how many years of experience he has using GCC so that he can operate GCC according to specific user type.

Nick has 3 options to select:

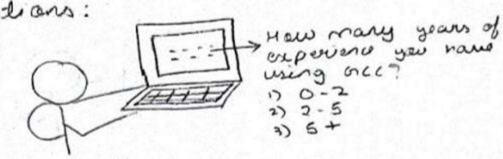
1) 0-2 years (For Novice programmer)
2) 2-5 years (For Typical programmer)
3) 5+ years (For Expert programmer)

Nick selects first option 0-2 years, as he is new to GCC. For every category, there is a set of commands which come in commands drop-down menu. Nick selects the first command from the commands drop-down menu.

After selecting the command, it appears in the input area. For compilation, Nick executes that command by clicking on the execute button. The result is visible in the output area. After using all the commands from the command list, Nick wants to explore all the commands form all commands menu available on top of the toolbar. He also feels at ease by selecting the command he executed previously from recent commands pane. After exploring all the commands Nick feels confident in programming and happy to use Smart GCC.

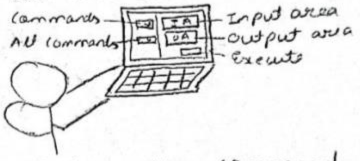
1) Nick Jones wants to learn Smart GCC.

2) When he opens Smart GCC, he has to choose one of the given options:

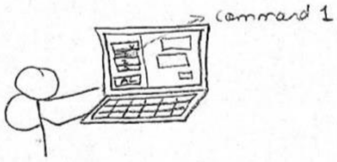


3) Nick is new to GCC, so he selects first option 0-2.

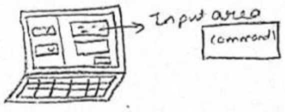
4) On the basis of 0-2 category, Nick comes under Novice programmer and for him, he has particular set of commands to use. But there is are more option to see all the commands.



5) Nick Selects first command from commands drop down menu.

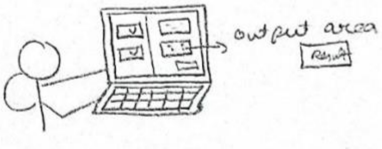


6) After selecting the command, it appears on input area.

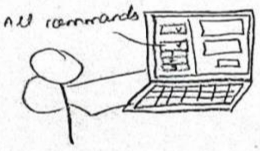


7) Now Nick wants to compile the command so he clicks on Execute button.

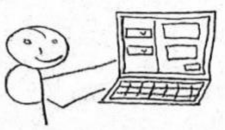
8) After compilation, result is produced in output area.



9) After using all the commands from command drop down menu, Nick wants to execute other commands from All commands list.



10) After exploring all the commands from All commands drop down menu, Nick feels confident in programming.



Prototype 2

Paper Based Prototype (wireframes on paper)

1. Usability of a Design

   a) Usability Goals:

- **Learnability:** Mental model is used to achieve this type of goal. It depends on how user learns new things.
- **Effectiveness:** The interface itself should guide the user to perform the required operations by having appropriate widgets with labels.
- **Utility:** Provides the necessary features which makes the user's tasks easy or fun to use.
- **Efficiency:** Makes the interface more intuitive and should help the user perform the tasks faster.
- **Memorability:** It is easier for users to remember how to carry out a task due to the mental model.

   b) Usability Principles:

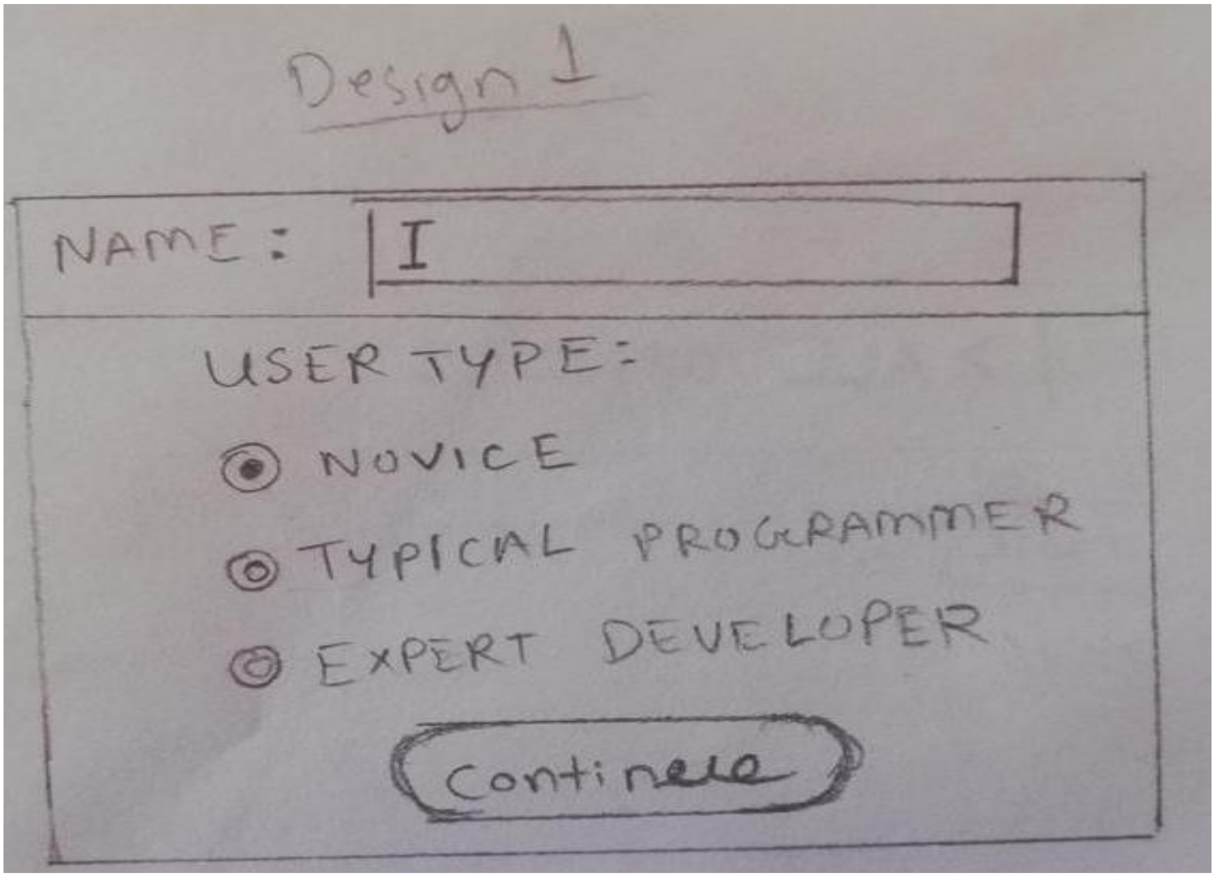
- **Consistency:** The app is consistent in terms of UI template throughout the application.
- **Visibility:** UI elements and icons which makes its functionality visible/obvious to the user. Ex.: For an error, there should be a popup with the error and a red stop sign.
- **Constraints:** Limits the functionalities according to the level of user or user type.
- **Affordance:** Usage of appropriate UI elements which denotes how to use it to perform the functionality.

2. Error Tolerance of the design

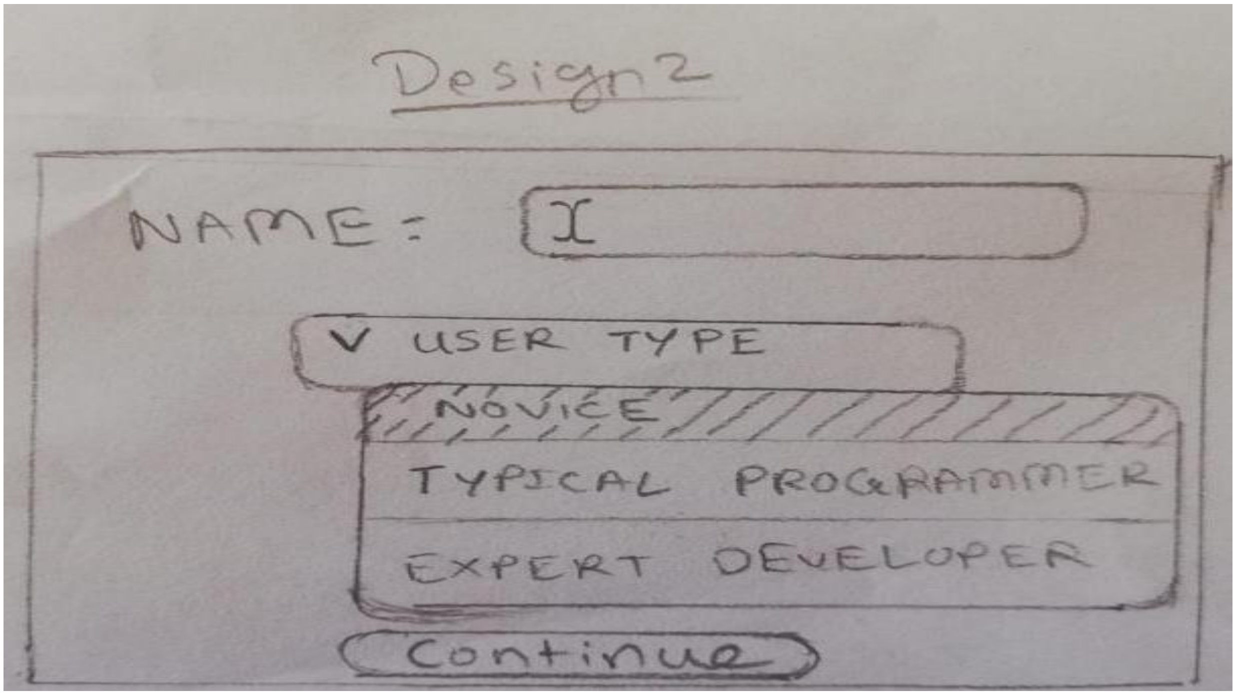
When user enters incorrect combination of GCC command and linking options, the user is prompted with error message and a stack trace on the console.

3. Efficiency of the design

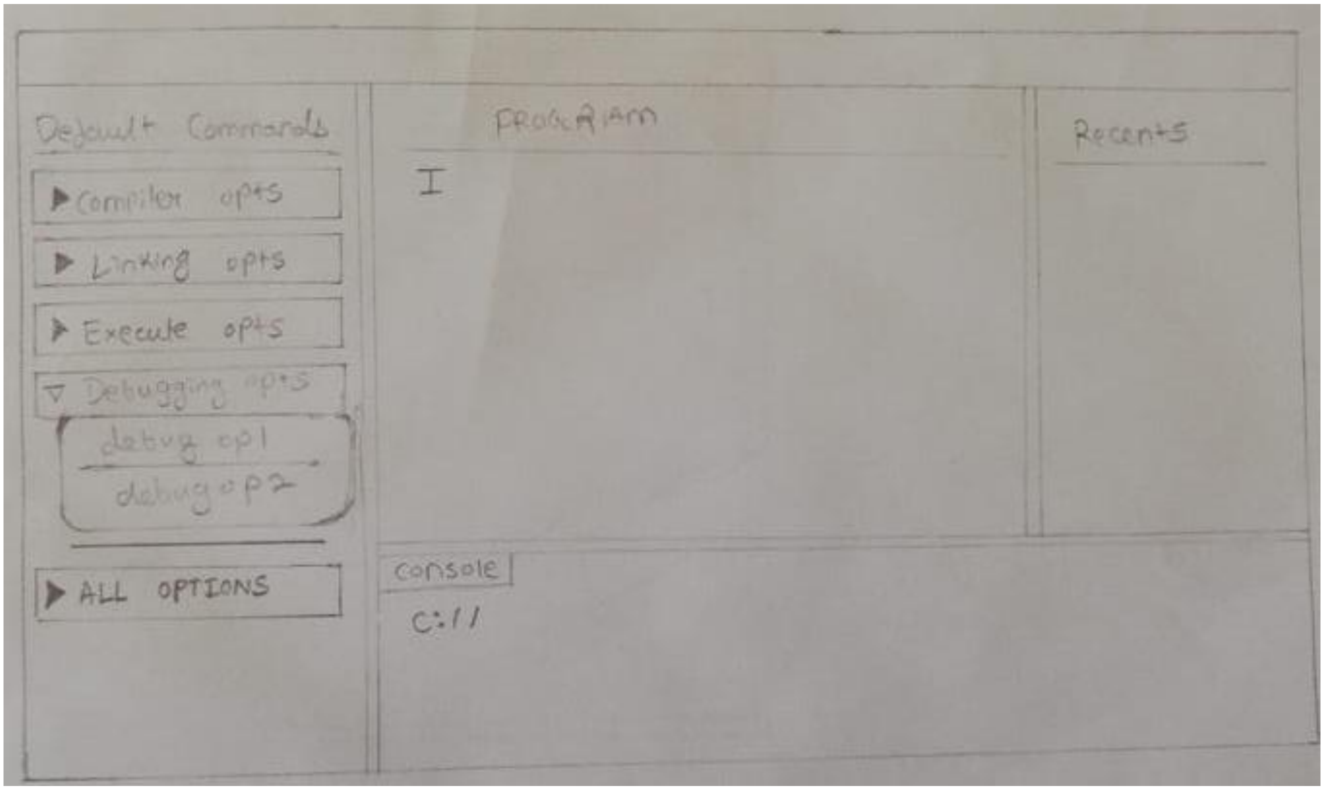
The UI is made simple by consolidating numerous options by categorising them and using appropriate widgets/ UI elements. The simplicity of the design helps provide a mental model to the user which makes the usage of software more efficient.



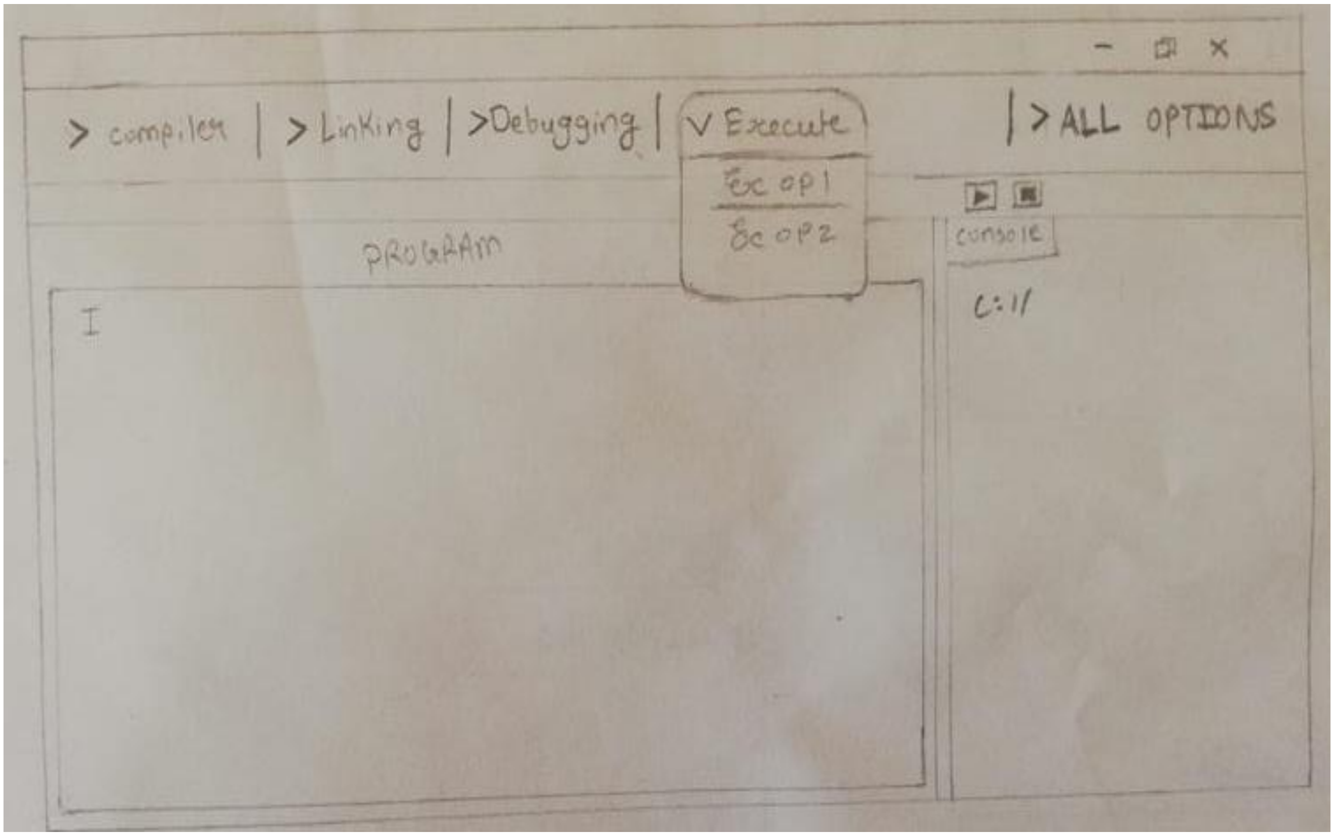
Selection of User Type screen design 1



Selection of User Type screen design 2



Main Screen Design Option 1



Main Screen Design Option 2

Responsibilities & Individual Contributions

Components	**Team Members**
Design 1 and Prototype 1	Yash Golwala, Chirag Vora, Harsh Divecha, Sanjana Udar, Pooja Dhir
Design 2 and Prototype 2	Harsh Divecha, Sanjana Udar, Chirag Vora, Yash Golwala, Jaswanth Banavathu
Design 3	Yash Golwala, Harsh Divecha, Chirag Vora, Sanjana Udar

References

[1] "Free prototyping tool for web & mobile apps - Justinmind", Justinmind.com, 2020. Available: https://www.justinmind.com/.

[2] R. Jayakumar, "Lecture Slides Requirements Gathering; Design, Prototyping and Construction".

[3] "Prototyping 101: The Difference between Low-Fidelity and High-Fidelity Prototypes and When to Use Each | Adobe Blog", Adobe Blog, 2020. [Online]. Available: https://theblog.adobe.com/prototyping-difference-low-fidelity-high-fidelity-prototypes-use/. [Accessed: 01- Mar- 2020].