****

**COMSATS University, Islamabad Pakistan**

**React Code Catalyzer**

***By***

**Asad Imtiaz Rana CIIT/SP18-BCS-031/ISB**

**Varda Quraishi CIIT/SP18-BCS-169/ISB**

***Supervisor*Dr. Adeel Anjum**

***Bachelor of Science in Computer Science / Software Engineering (2018-2022)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

****

**COMSATS University, Islamabad Pakistan**

**React Code Catalyzer**

**A project presented to**

**COMSATS University, Islamabad**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Computer Science / Software Engineering (2018-2022)***

**By**

**Asad Imtiaz Rana CIIT/SP18-BCS-031/ISB**

**Varda Quraishi CIIT/SP18-BCS-169/ISB**

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other, we will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Asad Imtiaz Rana Varda Quraishi

--------------------------- ---------------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS) “Project title” was developed by   
**Asad Imtiaz Rana (CIIT/SP18-BCS-031)** and **Varda Quraishi (CIIT/SP18 -BCS-169)** under the supervision of “Dr. Adeel Anjum” and co supervisor “CO-SUPERVISOR NAME” and that in (their/his/her) opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences / Software Engineering.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Head of Department**

**(Department of Computer Science)**

**Executive Summary**

RCC is a mobile application building software, with less code to no code building options. Usually, mobile application developers waste a lot of time in developing a whole app from scratch and if they face any errors, it is extremely tough for developers to sort it out. Updating the gradle files is also tiresome and time consuming. Our platform i.e., React Code Catalyzer will solve the above issues.

There will be tons of useful components that can be used by the developer for his/her application. Once a component is chosen, user can simply update all its props by adding the values in the text field of props. The user can either choose coding mode or drag and drop mode whatever he/she desires.

User can either build an app from built-in templates or showcase his/her own creativity by building apps from scratch.

RCC will cater the need of users interested in building apps of medium complexity commonly lying in the following categories: Lifestyle, social media, Utility, News/Information Outlets and Education.

To achieve these objectives developers of RCC have added modules; manage projects, manage store, app development, git-hub integration, integrated terminal, and automated template generator.

**Acknowledgement**

All praise is to Almighty Allah the Beneficial and the Merciful.

With the help of Almighty Allah, our industrious teachers who have imparted us knowledge that we have implemented in RCC. Our esteemed supervisor **Dr. Adeel Anjum** for always being our support and encouragement. Our parents who have always backed us to see us excel in life.

We have successfully projected RCC.

We are thankful to all of you from the depth of our hearts.

|  |  |  |
| --- | --- | --- |
| Asad Imtiaz Rana |  | Varda Quraishi |
| --------------------------- |  | --------------------------- |

**Abbreviations**

|  |  |
| --- | --- |
| **SRS** | Software Requirement Specification |
| **SDS** | Software Design Document |
| **RCC** | React Code Catalyzer |
| **PC** | Personal Computer |
| **MAD** | Mobile Application Development. |
| **WT** | Web Technologies |
| **ICT** | Introduction to Computer. |
| **OOP** | Object Oriented Programming |
| **PF** | Programming Fundamentals |
| **SW** | Software |
| **FB** | Facebook |
| **DD** | Drag N Drop |
| **AI** | Artificial Intelligence |
| **IDS** | Introduction to Data Science |

Contents

[1 Introduction 14](#_Toc89893056)

[1.1 Vision Statement 14](#_Toc89893057)

[1.2 Related System Analysis/Literature Review 14](#_Toc89893058)

[1.2.1 Interactive App Builder 14](#_Toc89893059)

[1.2.2 Appy Pie 15](#_Toc89893060)

[1.3 Project Deliverables 16](#_Toc89893061)

[1.4 System Limitations/Constraints 16](#_Toc89893062)

[1.5 Tools and Technologies 16](#_Toc89893063)

[1.6 Relevance to Course Modules 17](#_Toc89893064)

[2 Problem Definition 18](#_Toc89893065)

[2.1 Problem Statement 18](#_Toc89893068)

[2.2 Problem Solution 18](#_Toc89893069)

[2.3 Objectives of the Proposed System 18](#_Toc89893070)

[2.4 Scope 18](#_Toc89893071)

[2.5 Modules 19](#_Toc89893072)

[2.5.1 Module 1: Manage Projects 19](#_Toc89893073)

[2.5.2 Create new Project: 19](#_Toc89893074)

[2.5.3 Open an existing project: 19](#_Toc89893075)

[2.5.4 Drag n Drop mode: 19](#_Toc89893076)

[2.5.5 Coding mode: 19](#_Toc89893077)

[2.5.6 Toggle between coding mode and drag n drop mode: 19](#_Toc89893078)

[2.5.7 Real Time saving: 19](#_Toc89893079)

[2.6 Module 2: Manage Store (Components/Templates) 19](#_Toc89893080)

[2.6.1 Downloading Templates: 20](#_Toc89893081)

[2.6.2 Downloading Components: 20](#_Toc89893082)

[2.7 Module 3: App Development 20](#_Toc89893083)

[2.7.1 Screen Building: 20](#_Toc89893084)

[2.8 Module 4: GitHub Integration 20](#_Toc89893085)

[2.8.1 Clone: 20](#_Toc89893086)

[2.8.2 Pull: 20](#_Toc89893087)

[2.8.3 Add: 20](#_Toc89893088)

[2.8.4 Commit: 20](#_Toc89893089)

[2.8.5 Push: 21](#_Toc89893090)

[2.9 Module 5: Integrated Terminal 21](#_Toc89893091)

[2.9.1 Git Commands 21](#_Toc89893092)

[2.9.2 Windows native commands 21](#_Toc89893093)

[3 Requirement Analysis 21](#_Toc89893094)

[3.1 User classes and characteristics 21](#_Toc89893096)

[3.2 Requirement Identifying Technique 22](#_Toc89893097)

[3.2.1 Use Case Diagram 22](#_Toc89893098)

[3.2.2 Detailed Use Case 22](#_Toc89893099)

[3.3 Functional Requirements 40](#_Toc89893100)

[3.3.1 Functional Requirement for Sign Up 40](#_Toc89893101)

[3.3.2 Functional Requirement for Sign In 43](#_Toc89893102)

[3.3.3 Functional Requirement for Log Out 44](#_Toc89893103)

[3.3.4 Functional Requirement for Project 44](#_Toc89893104)

[3.3.5 Functional Requirement for Modes 46](#_Toc89893105)

[3.3.6 Functional Requirement for preserving and updating work 48](#_Toc89893106)

[3.3.7 Functional Requirement for selecting from store 49](#_Toc89893107)

[3.3.8 Functional Requirement for screen building 52](#_Toc89893108)

[3.3.9 Functional Requirement for navigation 54](#_Toc89893109)

[3.3.10 Functional Requirement for terminal commands 64](#_Toc89893110)

[3.3.11 Functional Requirement for version controlling 69](#_Toc89893111)

[3.4 Non-Functional Requirements 72](#_Toc89893112)

[3.4.1 Reliability 72](#_Toc89893113)

[3.4.2 Usability 72](#_Toc89893114)

[3.4.3 Performance 72](#_Toc89893115)

[3.4.4 Security 72](#_Toc89893116)

[3.5 External Interface Requirements 72](#_Toc89893117)

[3.5.1 User Interfaces Requirements 72](#_Toc89893118)

[3.5.2 Software interfaces 74](#_Toc89893119)

[3.5.3 Hardware interfaces 74](#_Toc89893120)

[3.5.4 Communications interfaces 74](#_Toc89893121)

[4 Design and Architecture 75](#_Toc89893122)

[4.1 Architectural Design 75](#_Toc89893127)

[4.2 Design Models 75](#_Toc89893128)

[4.2.1 Activity Diagram 75](#_Toc89893129)

[4.2.2 Data flow diagram 84](#_Toc89893130)

[4.2.3 Behavioral Diagram 86](#_Toc89893131)

[4.3 Data Design 87](#_Toc89893132)

[4.3.1 Data Dictionary 88](#_Toc89893133)

[4.4 Human Interface Design 89](#_Toc89893134)

[4.4.1 Screen Images 89](#_Toc89893135)

[4.5 Screen Objects and Actions 90](#_Toc89893136)

[5 Implementation 90](#_Toc89893137)

[5.1 Algorithm 90](#_Toc89893139)

[5.2 User Interface 93](#_Toc89893140)

[5.3 Deployment 98](#_Toc89893141)

[6 Testing and Evaluation 98](#_Toc89893142)

[6.1 Unit Testing 98](#_Toc89893145)

[6.2 Functional Testing 106](#_Toc89893146)

[6.3 Business Rules Testing 112](#_Toc89893147)

[6.4 Integration Testing 113](#_Toc89893148)

[7 Conclusion and Future Work 115](#_Toc89893149)

[7.1 Conclusion 115](#_Toc89893151)

[7.2 Future Work 115](#_Toc89893152)

[8. References 116](#_Toc89893153)

List of figures

[Figure 1 Interactive App Builder 14](#_Toc89892893)

[Figure 2 Appy Pie 15](#_Toc89892894)

[Figure 3 : User- Use Case Diagram for RCC 22](#_Toc89892895)

[Figure 4: Line-box 75](#_Toc89892896)

[Figure 5 Activity Diagram for Initializing Project. 76](#_Toc89892897)

[Figure 6 Activity Diagram for Open an existing Project. 76](#_Toc89892898)

[Figure 7 Activity Diagram for Toggle Modes. 77](#_Toc89892899)

[Figure 8 Activity Diagram for Save Projects. 77](#_Toc89892900)

[Figure 9 Activity Diagram for Access Store of templates. 78](#_Toc89892901)

[Figure 10 Activity Diagram for Access Store of components. 78](#_Toc89892902)

[Figure 11 Activity Diagram for Manage Component. 79](#_Toc89892903)

[Figure 12 Activity Diagram for Integrate Third Party Tools. 79](#_Toc89892904)

[Figure 13 Activity Diagram for Managing Navigation. 80](#_Toc89892905)

[Figure 14 Activity Diagram for Stack Navigation. 80](#_Toc89892906)

[Figure 15 Activity Diagram for Tab Navigation. 81](#_Toc89892907)

[Figure 16 Activity Diagram for Screen Building. 81](#_Toc89892908)

[Figure 17 Activity Diagram for Interacting with the Terminal. 82](#_Toc89892909)

[Figure 18 Activity Diagram for Running Windows Command. 82](#_Toc89892910)

[Figure 19 Activity Diagram for Running Git Command. 83](#_Toc89892911)

[Figure 20 Activity Diagram for Running NPM Command. 83](#_Toc89892912)

[Figure 21 Activity Diagram for Performing Version Control. 84](#_Toc89892913)

[Figure 22 Data Flow Diagram of Level 0. 84](#_Toc89892914)

[Figure 23 Data Flow Diagram of Level 1 85](#_Toc89892915)

[Figure 24 Data Flow Diagram of Level 2a. 85](#_Toc89892916)

[Figure 25 Data Flow Diagram of Level 2b. 86](#_Toc89892917)

[Figure 26 Control flow diagram depicting working of RCC 87](#_Toc89892918)

[Figure 30 Drag N Drop mode from user's perspective 89](#_Toc89892919)

[Figure 31 Coding mode from user's perspective 90](#_Toc89892920)

[Figure 32: UI for project creation. 93](#_Toc89892921)

[Figure 33: UI for adding screens 94](#_Toc89892922)

[Figure 34: UI for drag N drop mode. 94](#_Toc89892923)

[Figure 35: UI for coding mode. 95](#_Toc89892924)

[Figure 36: UI for git commit command 96](#_Toc89892925)

[Figure 37: UI for git clone command 96](#_Toc89892926)

[Figure 38: UI for git creating new branch command 97](#_Toc89892927)

List of Tables

[Table 1 Related System Analysis with proposed project solution 15](#_Toc89892928)

[Table 2: Textual Description of Signup. 23](#_Toc89892929)

[Table 3: Textual Description of Verify Email. 24](#_Toc89892930)

[Table 4: Textual Description of Login. 24](#_Toc89892931)

[Table 5: Textual Description of Logout. 25](#_Toc89892932)

[Table 6: Textual Description of Initializing Project. 26](#_Toc89892933)

[Table 7: Textual Description of Open an existing project. 27](#_Toc89892934)

[Table 8: Textual Description of Toggling between screens. 27](#_Toc89892935)

[Table 9: Textual Description of Real Time Saving. 28](#_Toc89892936)

[Table 10: Textual Description of Marketplace. 29](#_Toc89892937)

[Table 11: Textual Description of Templates. 30](#_Toc89892938)

[Table 12: Textual Description of Components. 30](#_Toc89892939)

[Table 13: Textual Description of Screen Building. 31](#_Toc89892940)

[Table 14: Textual Description of Navigation. 32](#_Toc89892941)

[Table 15: Textual Description of Drawer Navigator. 33](#_Toc89892942)

[Table 16: Textual Description of Stack Navigator. 34](#_Toc89892943)

[Table 17: Textual Description of Tab Navigator. 34](#_Toc89892944)

[Table 18: Textual Description of Integrating Third Party Tools. 35](#_Toc89892945)

[Table 19: Textual Description of Interacting with Terminal. 36](#_Toc89892946)

[Table 20: Textual Description Running Window’s native Commands. 37](#_Toc89892947)

[Table 21: Textual Description of Running Git Commands. 38](#_Toc89892948)

[Table 22: Textual Description of Running NPM Commands. 38](#_Toc89892949)

[Table 23: Textual Description of Running Git Commands. 39](#_Toc89892950)

[Table 24: Textual Description of GitHub Integration. 40](#_Toc89892951)

[Table 25: Description of FR-1 41](#_Toc89892952)

[Table 26: Description of FR-2 41](#_Toc89892953)

[Table 27: Description of FR-3 41](#_Toc89892954)

[Table 28: Description of FR-4 42](#_Toc89892955)

[Table 29: Description of FR-5 42](#_Toc89892956)

[Table 30: Description of FR-6 43](#_Toc89892957)

[Table 31: Description of FR-7 43](#_Toc89892958)

[Table 32: Description of FR-8 44](#_Toc89892959)

[Table 33: Description of FR-9 44](#_Toc89892960)

[Table 34: Description of FR-10 45](#_Toc89892961)

[Table 35: Description of FR-11 45](#_Toc89892962)

[Table 36: Description of FR-12 45](#_Toc89892963)

[Table 37: Description of FR-13 46](#_Toc89892964)

[Table 38: Description of FR-14 46](#_Toc89892965)

[Table 39: Description of FR-15 47](#_Toc89892966)

[Table 40: Description of FR-16 47](#_Toc89892967)

[Table 41: Description of FR-17 48](#_Toc89892968)

[Table 42: Description of FR-18 48](#_Toc89892969)

[Table 43: Description of FR-19 48](#_Toc89892970)

[Table 44: Description of FR-20 49](#_Toc89892971)

[Table 45: Description of FR-21 49](#_Toc89892972)

[Table 46: Description of FR-22 50](#_Toc89892973)

[Table 47: Description of FR-23 50](#_Toc89892974)

[Table 48: Description of FR-24 50](#_Toc89892975)

[Table 49: Description of FR-25 51](#_Toc89892976)

[Table 50: Description of FR-26 51](#_Toc89892977)

[Table 51: Description of FR-27 52](#_Toc89892978)

[Table 52: Description of FR-28 52](#_Toc89892979)

[Table 53: Description of FR-29 53](#_Toc89892980)

[Table 54: Description of FR-30 53](#_Toc89892981)

[Table 55: Description of FR-31 53](#_Toc89892982)

[Table 56: Description of FR-32 54](#_Toc89892983)

[Table 57: Description of FR-33 54](#_Toc89892984)

[Table 58: Description of FR-34 55](#_Toc89892985)

[Table 59: Description of FR-35 55](#_Toc89892986)

[Table 60: Description of FR-36 55](#_Toc89892987)

[Table 61: Description of FR-37 56](#_Toc89892988)

[Table 62: Description of FR-38 56](#_Toc89892989)

[Table 63: Description of FR-39 57](#_Toc89892990)

[Table 64: Description of FR-40 57](#_Toc89892991)

[Table 65: Description of FR-41 57](#_Toc89892992)

[Table 66: Description of FR-42 58](#_Toc89892993)

[Table 67: Description of FR-43 58](#_Toc89892994)

[Table 68: Description of FR-44 59](#_Toc89892995)

[Table 69: Description of FR-45 59](#_Toc89892996)

[Table 70: Description of FR-46 59](#_Toc89892997)

[Table 71: Description of FR-47 60](#_Toc89892998)

[Table 72: Description of FR-48 60](#_Toc89892999)

[Table 73: Description of FR-49 60](#_Toc89893000)

[Table 74: Description of FR-50 61](#_Toc89893001)

[Table 75: Description of FR-51 61](#_Toc89893002)

[Table 76: Description of FR-52 62](#_Toc89893003)

[Table 77: Description of FR-53 62](#_Toc89893004)

[Table 78: Description of FR-54 62](#_Toc89893005)

[Table 79: Description of FR-55 63](#_Toc89893006)

[Table 80: Description of FR-56 63](#_Toc89893007)

[Table 81: Description of FR-57 64](#_Toc89893008)

[Table 82: Description of FR-58 64](#_Toc89893009)

[Table 83: Description of FR-59 64](#_Toc89893010)

[Table 84: Description of FR-60 65](#_Toc89893011)

[Table 85: Description of FR-61 65](#_Toc89893012)

[Table 86: Description of FR-62 66](#_Toc89893013)

[Table 87: Description of FR-63 66](#_Toc89893014)

[Table 88: Description of FR-64 66](#_Toc89893015)

[Table 89: Description of FR-65 67](#_Toc89893016)

[Table 90: Description of FR-66 67](#_Toc89893017)

[Table 91: Description of FR-67 67](#_Toc89893018)

[Table 92: Description of FR-68 68](#_Toc89893019)

[Table 93: Description of FR-69 68](#_Toc89893020)

[Table 94: Description of FR-70 69](#_Toc89893021)

[Table 95: Description of FR-91 69](#_Toc89893022)

[Table 96: Description of FR-92 70](#_Toc89893023)

[Table 97: Description of FR-93 70](#_Toc89893024)

[Table 98: Description of FR-94 70](#_Toc89893025)

[Table 99: Description of FR-95 71](#_Toc89893026)

[Table 100: Description of FR-96 71](#_Toc89893027)

[Table 101 Core algorithm implementation table. 90](#_Toc89893028)

[Table 102: Test Cases for Managing App 98](#_Toc89893029)

[Table 103: Test Cases for Managing Modes 99](#_Toc89893030)

[Table 104: Test Cases for Auto Save 99](#_Toc89893031)

[Table 105: Test Cases for Downloading Components 100](#_Toc89893032)

[Table 106: Test Cases for Downloading Templates 100](#_Toc89893033)

[Table 107: Test Cases for Adding More than 1 screen 101](#_Toc89893034)

[Table 108: Test Cases for Stack Navigation 101](#_Toc89893035)

[Table 109: Test Cases for Tab Navigation 101](#_Toc89893036)

[Table 110: Test Cases for cloning from GitHub 102](#_Toc89893037)

[Table 111: Test Cases for pulling from GitHub 102](#_Toc89893038)

[Table 112: Test Cases for pushing to GitHub 103](#_Toc89893039)

[Table 113: Test Cases for adding to GitHub 103](#_Toc89893040)

[Table 114: Test Cases for committing to GitHub 103](#_Toc89893041)

[Table 115: Test Cases for Terminal 104](#_Toc89893042)

[Table 116: Test Cases for Terminal 104](#_Toc89893043)

[Table 117: Test Cases for Terminal 105](#_Toc89893044)

[Table 118: Test Cases for Terminal 105](#_Toc89893045)

[Table 119 Test for drawer navigation 106](#_Toc89893046)

[Table 120 Test script for Module 1-Managing Projects 106](#_Toc89893047)

[Table 121 Test Script for component testing. 107](#_Toc89893048)

[Table 122 Test script for Module-3 Developing the app 110](#_Toc89893049)

[Table 123 Test script for Module-4 performing version controlling 111](#_Toc89893050)

[Table 124 Test script for Module-5 Integrated Terminal 111](#_Toc89893051)

[Table 125 Test Script for Project Management 113](#_Toc89893052)

[Table 126 Test Script for Interaction between modes and marketplace 114](#_Toc89893053)

[Table 127 Test Script for Screen Navigation. 114](#_Toc89893054)

[Table 128 Test Script for interaction of components with screen 115](#_Toc89893055)

# Introduction

Usually, mobile application developers waste a lot of time in developing a whole app from scratch and if they face any errors, it is extremely tough for developers to sort it out and updating gradle files is tiresome and time consuming. Our platform i.e., React Code Catalyzer will solve this issue. There will be tons of useful components that can be used by developer for his application. Once a component is chosen, user can simply update all its props by simply adding the values in the text field of props.

## Vision Statement

RCC is a desktop application with two modes, drag n drop and coding, build for mobile application developers and entrepreneurs who have established businesses and need an application to cater their business needs. RCC will aid users to make react native applications with just drag and drop with no coding or extensive debugging. Users can experiment between themes and see what suits the best to them. For mobile developers this tool can come in handy in places where they cater a bug; they can simply switch from coding mode to the drag n drop mode and perform their respective tasks. In this digitized world a need for an application is inevitable.

Unlike the current systems which are web based and always need an active internet to work RCC is desktop based and internet is only required when downloading any component. The related systems have limited layouts to choose from and the user needs to buy additional services to generate an APK. Our product is free unlike others. The code will be on the users’ machine so build will not take too much.

## Related System Analysis/Literature Review

### Interactive App Builder

This is a web-based editor, that always needs an active internet to work. The software can build react native mobile applications with drag and drop. The manufacturers of the system have released the beta version of it till date.

A picture containing text, electronics, display, screenshot

Description automatically generated

Figure Interactive App Builder

### Appy Pie

This is also a web-based editor, that always needs an active internet to work. The software provides various kind of services, building mobile applications, websites, task automation, chatbot, graphics and live chat. It has some selected number of layouts to choose from.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure Appy Pie

The table below shows the weaknesses of already functional systems and how RCC proves to be better than the former systems.

Table Related System Analysis with proposed project solution

|  |  |  |
| --- | --- | --- |
|  | **Weakness** | **Proposed Project Solution** |
| Interactive App Builder | It always needs internet to work as it is web-based editor.  The whole code is on server, so the build takes a lot of time.  You need to buy additional services to generate APK. | We will develop desktop app, so it will not always require internet to work.  The code will be on user’s machine so the build will not take much time and resources.  User will have control over his code so he can generate APK as much times as he wants. |
| Appy Pie | Appy Pie needs internet to work as it is web-based editor.  Appy Pie allows us to create a limited number of applications for free.  Appy Pie has limited number of layouts. | We will develop desktop app, so it will not always require internet to work.  Our system will always be free if you want to create applications from scratch or use our free themes.  Our system will have unlimited number of free and paid themes as a developer can create a theme and then upload it to our marketplace. |

## Project Deliverables

List down the project deliverables.

Scope document.

Software Requirement Specification document.

Software Design Specification document.

Test Plan document.

Final Report

## System Limitations/Constraints

LI-1: User will need internet to interact our system with the package manager.

LI-2: User will need a macOS or iPhone to test iOS application.

## Tools and Technologies

Below are all the hardware/software tools and technologies with version number which were used in implementation of the project.

Table 3 Tools and Technologies for Proposed Project

|  |  |  |  |
| --- | --- | --- | --- |
| **Tools**  **And**  **Technologies** | **Tools** | **Version** | **Rationale** |
| Visual Studio Code | 1.51 | IDE |
| Firebase | 8. | Data Store |
| Adobe XD | CC 2018 | Design Work |
| MS Word | 2019 | Documentation |
| MS Power Point | 2019 | Presentation |
| MS Project Pro | 2019 | Gantt Chart Creation |
| Google Forms | 2017 | Requirements Gathering Survey |
| Creatly | 7.0.15 | Diagrams |
| MS Visio | 2019 | Diagram(s) |
| Technology | **Version** | **Rationale** |
| Electron JS | 11.2.0 | Desktop App Development |
| React JS | 17.0.0 | Front-End Programming Language |
| React Native | 0.63.0 | Mobile App Development |
| Node JS | 15.3.0 | Back-End Programming Language |
| HTML | 5 | Markup Language |
| CSS | 3 | Styling HTML Document |
| JavaScript | ES6 | Programming Language |

## Relevance to Course Modules

Firstly, the basic idea of RCC came into our minds after encountering difficulty in creating application during our course of MAD.

We have used react js which we had learned during our course of WT.

Working with databases was taught during Database 1.

A brief code of python is used for web scrapping, this language was taught in earlier semesters during the course of ICT, AI, IDS.

The working methodology was easy to select as programming methodologies were taught during the course of OOP and PF.

# Problem Definition

Cater the difficulties of react native mobile app developers by providing them an alternate solution to build applications with drag n drop.



## Problem Statement

It is difficult to create mobile applications for small to medium scale startups as they have low budgets. RCC will aid them in building applications for their businesses. Even the best app idea from a concept to a product is expensive, time consuming and labor extensive. Less to no code development can increase the production cost with better quality.

As for the developer, needs a lot of time to develop a mobile application by writing a lot of code and debugging it. At times developers get stuck at places, there RCC will come in handy.

Lastly systems already built do not cater the need of users who are interested in toggling between coding mode and drag and drop mode.

## Problem Solution

Building applications with RCC requires lesser resources. Therefore, reducing production cost of mobile applications. With using RCC there is no need to hire developers as this application is user friendly and does not need high level knowledge of React Native. Anyone can build applications using RCC.

## Objectives of the Proposed System

BO-1: Reduce the cost of producing mobile applications.

BO-2: Reduce the production time.

BO-3: Experiment yourself, no need to have a high-level knowledge of react.

BO-4: Keep updating with time without having to pay.

BO-5: Switch from development to drag N drop where programmer gets stuck.

## Scope

The system will be publicly available as soon as it is released, and it will support windows as well as mac. It will be free to download and free to use by anyone at any place in the world. Initially, our system will need an email account to sign up and an internet connection to download the application. This is one-time process.

After the download is finished, users will need to install the application which will not need an internet connection. After the installation is finished, users can create a new project, create a whole application from scratch or use our free or paid components or templates. Once the application is made, a user can also generate its build to release on google play store or apple app store. This whole app development process does not need an internet connection except when a user needs to install some dependencies from the terminal through npm or download our templates through our own package manager.

## Modules

Below are all the modules of RCC.

### Module 1: Manage Projects

In App Initialization a user will be able to create a new project, open an existing project, develop application in our drag n drop mode, develop app by writing code, toggle between drag n drop mode or coding mode.

### Create new Project:

A new project will be created, and all its dependencies will be installed for android and iOS.

### Open an existing project:

Users can open an existing project that is developed by using our system and continue your work from where you left it.

### Drag n Drop mode:

Users can develop a whole application in drag n drop mode from scratch or by using themes and components from our marketplace.

### Coding mode:

Users can read, write, edit his/her code in coding mode and the changes will be reflected.

### Toggle between coding mode and drag n drop mode:

Users can switch between drag n drop mode and coding mode as per user’s convenience.

### Real Time saving:

This feature will allow the user to automatically save the code without pressing ctrl + S every time a new component is added in the app. So, your work will be automatically saved in case of power cutdown.

## Module 2: Manage Store (Components/Templates)

This module is added to make a developer’s life much easier by adding tons of templates and components.

### Downloading Templates:

Initially we will develop 1 template and after our system is completely developed, we will open our store to public where a developer can upload a personally customized template so that other developers can use it.

### Downloading Components:

Initially we will develop 10 components and after our system is completely developed, we will open our store to public where a developer can upload a personally customized component so that other developers can use it.

## Module 3: App Development

App development is the main module of our system where a user will be able to build screens.

### Screen Building:

This feature will allow users to build screens by drag n drop. A user can also view his code by toggling into coding mode.

## Module 4: GitHub Integration

This module will be used to clone, pull, add, commit, and push to GitHub repository. Hence our system also supports version control.

### Clone:

This feature will allow a user to clone his project to a GitHub repository.

### Pull:

This feature will allow a user to pull the code from existing GitHub repository.

### Add:

This feature will allow a user to add the changes in his project to store locally on his git.

### Commit:

This feature will allow a user to commit the changes in his git.

### Push:

This feature will allow a user to push his code to its GitHub repository.

## Module 5: Integrated Terminal

An integrated terminal will be developed that will be used to install packages from our own package manager and run windows native commands.

### Git Commands

Users can run git commands if he is more comfortable to use GitHub commands.

### Windows native commands

Users can run windows commands that are used in command prompt to create a file/folder navigate to a folder.

# Requirement Analysis

This part will cover the overall perspective of our project (RCC). It will describe all the requirement gathering techniques included to make RCC a user friendly and a beneficial software. The use case diagrams will cover how our user(developer) will interact with the system; what functionalities can the user perform will be demonstrated in tabular form under functional requirements.



## User classes and characteristics

RCC is a single user system. The table below shows the pertinent characteristics of the user.

Table 4 User classes and characteristics table

|  |  |
| --- | --- |
| **User class** | **Description** |
| User | The user can be any person that uses RCC to build any application. May it be a developer or an entrepreneur. The user can perform all functionalities that RCC provides. Initializing applications, building applications, accessing the store to download components and embed them in their system, interacting with terminal, performing version control. |

## Requirement Identifying Technique

This section describes the requirements identifying technique(s) which further help to derive functional requirements specification.

### Use Case Diagram

Diagram

Description automatically generated

Figure : User- Use Case Diagram for RCC

### Detailed Use Case

This table explains the use case diagram for sign up. Signup is required for user to keep track of user’s projects and upload a template to our marketplace.

Table : Textual Description of Signup.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-1 |
| **Use Case Name:** | Signup |
| **Actors:** | User |
| **Description:** | Signup is required for user to keep track of user’s projects and upload a template to our marketplace |
| **Trigger:** | It will automatically be triggered when a user visits our marketplace. |
| **Level:** | Low |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will enter username.  2. User will enter email.  3. User will enter password.  4. User will enter confirm password.  5. User will press signup button. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | 1. User did not enter a valid email  1.1 System indicates that the entered email is not a valid one.  1.2 System will prompt the user to re-enter email.  1.3 Return to step 2 of normal flow.  2. Password did not have 8 characters including 1 uppercase letter, 1 lower case letter and 1 digit  2.1 System indicates that the entered password is not a valid one.  2.2 System will prompt the user to re-enter password.  2.3 Return to step 3 of normal flow.  3. Password and confirm password field did not match.  3.1 System indicates that password and confirm password field did not match.  3.2 System will prompt the user to re-enter password and confirm password.  3.3 Return to step 3 of normal flow. |
| **Postconditions:** | [Describe the state of the system at the conclusion of the use case execution.  POST-1. System will prompt user to verify his email, this event will be triggered once. |
| **Business Rules** | N/A |
| **Assumptions:** | User should have an email. |

This table explains the use case diagram for verify email. Verify Email is required in order to further proceed with your account.

Table : Textual Description of Verify Email.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-2 |
| **Use Case Name:** | Verify Email |
| **Actors:** | User |
| **Description:** | Verify Email is required to further proceed with your account. |
| **Trigger:** | It will automatically be triggered when the user clicks on the signup button with valid credentials |
| **Level:** | Low |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will log into his/her email account.  2. User will open the relevant email.  3. User will click on verify now button. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | [Describe the state of the system at the conclusion of the use case execution.  POST-1. User will be redirected to Login page. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for login. Login is required for user to proceed with our marketplace.

Table : Textual Description of Login.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-3 |
| **Use Case Name:** | Login |
| **Actors:** | User |
| **Description:** | Login is required for user to proceed with our marketplace. |
| **Trigger:** | T1: It will automatically be triggered when user verifies email.  T2: It will be triggered with a button clicked on sign up page, named LOGIN. |
| **Level:** | Medium |
| **Preconditions:** | User has signed up.  User email verified. |
| **Includes:** | N/A |
| **Normal Flow:** | 1.1 User will input his/her email.  1.2 User will input password.  1.3 User will click on Login button.  2.1 User will click on the “continue with google” button.  2.2 User will select his email ID. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | [Describe the state of the system at the conclusion of the use case execution.  POST-1. User will be redirected to Login page. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for logout. Needed for user to log out from a specific account. Mainly needed if the user accessed his/her account from someone else’s system or if user needs to change account.

Table : Textual Description of Logout.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-4 |
| **Use Case Name:** | Logout |
| **Actors:** | User |
| **Description:** | Needed for user to log out from a specific account. Mainly needed if the user accessed his/her account from someone else’s system or if user needs to change account. |
| **Trigger:** | It will automatically be triggered when the user clicks on the signup button with valid credentials |
| **Level:** | Medium |
| **Preconditions:** | User logged in. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will press logout button. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | N/A |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for initializing project. The user will be able to create a new project.

Table : Textual Description of Initializing Project.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-5 |
| **Use Case Name:** | Initializing Project |
| **Actors:** | User |
| **Description:** | The user will be able to create a new project. |
| **Trigger:** | In the header there will be an option named File by which a dropdown will appear. In this dropdown there will be an option named “Create Project”. |
| **Level:** | High |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click create project.  2. A pop up window will pop.  3. User will enter project name.  4. User will enter, the path where the file will be saved.  5. User will click save button. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | 1.1 The project with same name already exists.  1.2 Return to step 2 of normal flow. |
| **Postconditions:** | A new empty project, draft screen will pop. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for open existing project. The user will be able to open an existing project that is created by using RCC.

Table : Textual Description of Open an existing project.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-6 |
| **Use Case Name:** | Open Existing Project |
| **Actors:** | User |
| **Description:** | The user will be able to open an existing project that is created by using RCC. |
| **Trigger:** | In the header there will be an option named File by which a dropdown will appear. In this dropdown there will be an option named “Open Project”. |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | UC-5 |
| **Normal Flow:** | 1. User will click open project.  2. A pop up window will pop, with a list of projects created lately.  3. User can either search a project name or select one from the list.  4. Click open. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The project opens on the screen. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for toggling between screens. This will enable user to switch mode.

Table : Textual Description of Toggling between screens.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-7 |
| **Use Case Name:** | Toggling between screens |
| **Actors:** | User |
| **Description:** | This will enable user to switch mode. |
| **Trigger:** | Inter mode switching will be triggered once the user selects UI or Code in the top bar. |
| **Level:** | Medium |
| **Preconditions:** | User has opened a project on the current screen. |
| **Includes:** | UC-5, UC-6 |
| **Normal Flow:** | 1. User will click on UI, to select drag and drop mode.  2. User will click Code, to select coding mode. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | 1.1 Project not opened on the current screen.  1.2 User will be prompted to first create a project. |
| **Postconditions:** | POST-1. Mode changed |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for real time saving. User saves his work without having to click save button or hot keys like ctrl + s.

Table : Textual Description of Real Time Saving.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-8 |
| **Use Case Name:** | Real Time Saving |
| **Actors:** | User |
| **Description:** | User saves his work without having to click save button or hot keys like ctrl + s. |
| **Trigger:** | Auto triggered, once there are changes detected. |
| **Level:** | Medium |
| **Preconditions:** | User has opened a project on the current screen.  User has written a line of code or performed any operation in the drag and drop mode. |
| **Includes:** | UC-5, UC-6 |
| **Normal Flow:** | 1. As soon as there are changes detected on the screen, the screen is auto saved. It be on any mode. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. Work saved. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for marketplace. User can download templates and components from our marketplace.

Table : Textual Description of Marketplace.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-9 |
| **Use Case Name:** | Marketplace |
| **Actors:** | User |
| **Description:** | User can upload templates and components to our marketplace.  User can download templates and components from our marketplace. |
| **Trigger:** | 1. There will be a third option parallel to UI/Code view where a user will have a marketplace option.  2. User can go to our online marketplace by entering our url in browser. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | UC-10, UC-11 |
| **Normal Flow:** | 1. User will click marketplace.  2. The screen will navigate to marketplace screen.  3.1 User will select his/her desired component.  3.2 User will click on download button on his selected component.  3.3 These components will be visible on the left side under download segment.  3.4 User will drag a component and drop it to the desired location on his application.  4.1 User will select his/her desired template.  4.2 User will click on use template button on his selected template.  4.3 These templates will be installed in the project and the changes will be reflected in the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | N/A |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for templates. There will be list of templates to choose from.

Table : Textual Description of Templates.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-10 |
| **Use Case Name:** | Templates |
| **Actors:** | User |
| **Description:** | There will be list of templates to choose from. |
| **Trigger:** | 1. There will be a third option parallel to UI/Code view where a user will have a marketplace option where there will be an option named templates.  2. User can go to our online marketplace by entering our url in browser and click on option named templates. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will go to our marketplace.  2. User will click on option named templates.  3. User will select his/her desired template.  4. User will click on use template button on his selected template.  5. The template will be installed and reflected on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The template will be installed and the changes will be reflected in the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for components. There will be list of components to choose from.

Table : Textual Description of Components.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-11 |
| **Use Case Name:** | Components |
| **Actors:** | User |
| **Description:** | There will be list of components to choose from. |
| **Trigger:** | 1. There will be a third option parallel to UI/Code view where a user will have a marketplace option where there will be an option named components.  2. User can go to our online marketplace by entering our url in browser and click on option named components. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will go to our marketplace.  2. User will click on option named components.  3. User will select his/her desired component.  4. User will click on download button on his selected component.  5. These components will be visible on the left side under download segment |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The component will appear on left under downloaded components segment |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for screen building. There will be a segment which will contain react-native’s default screen building components like View, Touchable Opacity etc.

Table : Textual Description of Screen Building.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-12 |
| **Use Case Name:** | Screen Building |
| **Actors:** | User |
| **Description:** | There will be a segment which will contain react-native’s default screen building components like View, Touchable Opacity etc. |
| **Trigger:** | On left bar there will be a segment which will have react-native’s default components to build screens. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click on a segment named Screen Building on left bar.  2. User will drag his desired component from left bar and drop it on the application.  3. The component will be added on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The component will be added to the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for navigation. There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc.

Table : Textual Description of Navigation.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-13 |
| **Use Case Name:** | Navigation |
| **Actors:** | User |
| **Description:** | There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. |
| **Trigger:** | On left bar there will be a segment which will have react-native’s default components for navigation. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | UC-14, UC-15, UC-16 |
| **Normal Flow:** | 1. User will click on a segment representing navigation on left bar.  2. User will drag his desired navigation component from this segment and drop it on the application.  3. The required navigation will be added on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The navigation will be added to the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for drawer navigation. There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Drawer Navigator and drop it to his application. The Drawer Navigator will be added in user’s application.

Table : Textual Description of Drawer Navigator.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-14 |
| **Use Case Name:** | Drawer Navigation |
| **Actors:** | User |
| **Description:** | There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Drawer Navigator and drop it to his application. The Drawer Navigator will be added in user’s application. |
| **Trigger:** | On left bar there will be a segment which will have react-native’s default components for navigation, in this segment there will be an option referring Drawer Navigator. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click on a segment representing navigation on left bar.  2. User will drag Drawer Navigator from this segment and drop it on the application.  3. The Drawer Navigation will be added on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The Stack Navigator will be added to the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for stack navigation. There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Stack Navigator and drop it to his application. The Stack Navigator will be added in user’s application.

Table : Textual Description of Stack Navigator.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-15 |
| **Use Case Name:** | Stack Navigation |
| **Actors:** | User |
| **Description:** | There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Stack Navigator and drop it to his application. The Stack Navigator will be added in user’s application. |
| **Trigger:** | On left bar there will be a segment which will have react-native’s default components for navigation, in this segment there will be an option reffering Stack Navigator. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click on a segment representing navigation on left bar.  2. User will drag Stack Navigator from this segment and drop it on the application.  3. The Stack navigator will be added on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The Stack Navigator will be added to the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for tab navigator. There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Tab Navigator and drop it to his application. The Tab Navigator will be added in user’s application.

Table : Textual Description of Tab Navigator.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-16 |
| **Use Case Name:** | Tab Navigator |
| **Actors:** | User |
| **Description:** | There will be a segment which will contain react-native’s default navigation components like Tab Navigator, Stack Navigator, Drawer Navigator etc. User will drag Tab Navigator and drop it to his application. The Tab Navigator will be added in user’s application. |
| **Trigger:** | On left bar there will be a segment which will have react-native’s default components for navigation, in this segment there will be an option reffering Tab Navigator. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click on a segment representing navigation on left bar.  2. User will drag Tab Navigator from this segment and drop it on the application.  3. The Tab Navigator will be added on the application. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The Tab Navigator will be added to the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for integrating 3rd party tools. There will be a segment representing third party tools integration. User will click on it which will show a list of third-party tools after clicking on a particular tool a new popup window will be triggered which will contain information about usage of a particular third-party tool. After performing required actions for a particular tool, it will be installed in our system.

Table : Textual Description of Integrating Third Party Tools.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-17 |
| **Use Case Name:** | Integrating Third Party Tools |
| **Actors:** | User |
| **Description:** | There will be a segment representing third party tools integration. User will click on it which will show a list of third-party tools after clicking on a particular tool a new popup window will be triggered which will contain information about usage of a particular third-party tool. After performing required actions for a particular tool, it will be installed in our system. |
| **Trigger:** | There will be a segment representing third party tools integration. User will click on it which will show a list of third-party tools. |
| **Level:** | Medium |
| **Preconditions:** | A project is created or opened. |
| **Includes:** | UC-18, UC-19 |
| **Normal Flow:** | 1. User will click on a segment representing third party tools integration on left bar.  2. User will click on it which will show a list of third-party tools.  3. User will click on a particular third-party tool which will trigger a popup screen.  4. User will perform the required actions and the required tool will be integrated |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The third party tool will be integrated in the application |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for interacting with terminal. An integrated terminal will be developed so that a user who is already more comfortable in working with terminal won’t feel missed out. User will be able to run git commands, window’s native commands or npm commands on it.

Table : Textual Description of Interacting with Terminal.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-20 |
| **Use Case Name:** | Interacting with Terminal |
| **Actors:** | User |
| **Description:** | An integrated terminal will be developed so that a user who is already more comfortable in working with terminal won’t feel missed out. User will be able to run git commands, window’s native commands or npm commands on it. |
| **Trigger:** | There will be an option named Terminal on top menu. |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | UC-21, UC-22, UC-23 |
| **Normal Flow:** | 1. User will click on Terminal option located on top menu.  2. A new terminal will popup.  3. User enters his desired command.  4. User presses enter.  5. The required command will be executed |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. The required command will be executed on Terminal. |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for windows command. User will be able to run window’s native commands on our terminal like mkdir, ls, cd, rm etc.

Table : Textual Description Running Window’s native Commands.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-21 |
| **Use Case Name:** | Running Window’s native Commands |
| **Actors:** | User |
| **Description:** | User will be able to run window’s native commands on our terminal like mkdir, ls, cd, rm etc. |
| **Trigger:** | Always Triggered when terminal is opened |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will enter window’s native command.  2. User will press enter.  3. The command will be executed. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. Window’s Native command will be executed |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for git commands. User will be able to run git commands on our terminal like clone, add, pull, push, branch etc.

Table : Textual Description of Running Git Commands.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-22 |
| **Use Case Name:** | Running Git Commands |
| **Actors:** | User |
| **Description:** | User will be able to run git commands on our terminal like clone, add, pull, push, branch etc. |
| **Trigger:** | Always Triggered when the terminal is opened |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will enter git command.  2. User will press enter.  3. The command will be executed. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. Git command will be executed |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for npm commands. User will be able to run npm commands on our terminal like install, uninstall, update etc.

Table : Textual Description of Running NPM Commands.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-23 |
| **Use Case Name:** | Running NPM Commands |
| **Actors:** | User |
| **Description:** | User will be able to run npm commands on our terminal like install, uninstall, update etc. |
| **Trigger:** | Always Triggered when the terminal is opened |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will enter npm command.  2. User will press enter.  3. The command will be executed. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. NPM command will be executed |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for User will be able to run npm commands on our terminal like install, uninstall, update etc.

Table : Textual Description of Running Git Commands.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-24 |
| **Use Case Name:** | Running NPM Commands |
| **Actors:** | User |
| **Description:** | User will be able to run npm commands on our terminal like install, uninstall, update etc. |
| **Trigger:** | Always Triggered when the terminal is opened |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will enter npm command.  2. User will press enter.  3. The command will be executed. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. NPM command will be executed |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

This table explains the use case diagram for GitHub integration. There will be buttons to clone, add, commit, and push your project to a GitHub repository for version control.

Table : Textual Description of GitHub Integration.

|  |  |
| --- | --- |
| **Use Case ID:** | UC-25 |
| **Use Case Name:** | GitHub Integration |
| **Actors:** | User |
| **Description:** | There will be buttons to clone, add, commit and push your project to a GitHub repository for version control |
| **Trigger:** | There will be 4 buttons on top bar of the screen to clone, add, commit or push your project to a GitHub repository. |
| **Level:** | Medium |
| **Preconditions:** | N/A |
| **Includes:** | N/A |
| **Normal Flow:** | 1. User will click on a required git button (clone, add, commit, push).  2. Corresponding git action will be performed (clone, add, commit, push). |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | N/A |
| **Exceptions:** | N/A |
| **Postconditions:** | POST-1. Git action will be performed (clone, add, commit, push) |
| **Business Rules** | N/A |
| **Assumptions:** | N/A |

## Functional Requirements

This section describes the functional requirements of RCC.

### Functional Requirement for Sign Up

Username is required to register the user against that respective name so that the user may be able to sign in with that name in future.

Table : Description of FR-1

|  |  |
| --- | --- |
| **Identifier** | FR-1 |
| **Title** | Username for signup |
| **Requirement** | The user shall be able to enter his/her name |
| **Source** | User |
| **Rationale** | Username is required to register the user against that respective name so that the user may be able to sign in with that name in future. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Email is required to keep track of user to suggest templates and components with respect to his recent searches.

Table : Description of FR-2

|  |  |
| --- | --- |
| **Identifier** | FR-2 |
| **Title** | User email for signup |
| **Requirement** | The user shall be able to enter his email |
| **Source** | User |
| **Rationale** | Email is required to keep track of user to suggest templates and components with respect to his recent searches |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user will enter a password containg 8 digits, with 1 uppercase, 1 lower case, 1 digit and 1 special character at least.

Table : Description of FR-3

|  |  |
| --- | --- |
| **Identifier** | FR-3 |
| **Title** | Password for sign up |
| **Requirement** | The user will enter a password containg 8 digits, with 1 uppercase, 1 lower case, 1 digit and 1 special character at least. |
| **Source** | User |
| **Rationale** | Password is required to maintain every user’s security. Maintain the system from unauthorized access. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

It is required to detect any errors of typo and to reconfirm the previously entered password.

Table : Description of FR-4

|  |  |
| --- | --- |
| **Identifier** | FR-4 |
| **Title** | Confirm Password for signup |
| **Requirement** | The user shall be able to re-enter the same password. |
| **Source** | User |
| **Rationale** | It is required in order to detect any errors of typo and to reconfirm the previously entered password. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-3 |
| **Priority** | High |

Verification of email is required to verify the user against his/her email. So that no one enters our webstore through someone else’s email. Verified emails will be kept in the database, to ensure single registration against single email.

Table : Description of FR-5

|  |  |
| --- | --- |
| **Identifier** | FR-5 |
| **Title** | User email verification link |
| **Requirement** | The user shall click on a link received after signup |
| **Source** | User |
| **Rationale** | Verification of email is required in order to verify the user against his/her email. So that no one enters our webstore through some one else’s email. Verified emails will be kept in the database, to ensure single registration against single email. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-1-4 |
| **Priority** | High |

### Functional Requirement for Sign In

Email is required to keep track of user to suggest templates and components with respect to his recent searches.

Table : Description of FR-6

|  |  |
| --- | --- |
| **Identifier** | FR-6 |
| **Title** | User email for signin |
| **Requirement** | The user shall be able to enter his email |
| **Source** | User |
| **Rationale** | Email is required to keep track of user to suggest templates and components with respect to his recent searches |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user will enter a password containg 8 digits, with 1 uppercase, 1 lower case, 1 digit and 1 special character at least.

Table : Description of FR-7

|  |  |
| --- | --- |
| **Identifier** | FR-7 |
| **Title** | Password for sign in |
| **Requirement** | The user will enter a password containg 8 digits, with 1 uppercase, 1 lower case, 1 digit and 1 special character at least. |
| **Source** | User |
| **Rationale** | Password is required to maintain every users security. Maintain the system from unauthorized access. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-6 |
| **Priority** | High |

The user shall be able to sign in with his/her google account as an alternative to signing in with username and password.

Table : Description of FR-8

|  |  |
| --- | --- |
| **Identifier** | FR-8 |
| **Title** | Signing In with Google Account |
| **Requirement** | The user shall be able to sign in with his/her google account as an alternative to signing in with username and password. |
| **Source** | User |
| **Rationale** | This is an easy sign in option for those having a google account. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Low |

### Functional Requirement for Log Out

Needed for user to log out from a specific account. Mainly needed if the user accessed his/her account from someone else’s system or if user needs to change account.

Table : Description of FR-9

|  |  |
| --- | --- |
| **Identifier** | FR-9 |
| **Title** | Logging out with the log out button |
| **Requirement** | The user shall be able to log out from the logged in account. |
| **Source** | User |
| **Rationale** | Needed for user to log out from a specific account. Mainly needed if the user accessed his/her account from someone elses system or if user needs to change account. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR- 6-8 |
| **Priority** | Medium |

### Functional Requirement for Project

Project Name is created to access a project further in future.

Table : Description of FR-10

|  |  |
| --- | --- |
| **Identifier** | FR-10 |
| **Title** | Entering project name |
| **Requirement** | The user shall be able to enter a name for his/her project. |
| **Source** | User |
| **Rationale** | Project Name is created to access a project further in future. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user shall be able to select a directory for the project on the local machine.

Table : Description of FR-11

|  |  |
| --- | --- |
| **Identifier** | FR-11 |
| **Title** | Selecting project directory |
| **Requirement** | The user shall be able to select a directory for the project on the local machine. |
| **Source** | User |
| **Rationale** | The purpose of this is to make it visible to the user where the project is being created in the local machine. Aiding in organizing the projects. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-10 |
| **Priority** | High |

Project Name is created to access a project further in future.

Table : Description of FR-12

|  |  |
| --- | --- |
| **Identifier** | FR-12 |
| **Title** | Updating Project Name |
| **Requirement** | The user shall be able to re-enter a name for his/her project. |
| **Source** | User |
| **Rationale** | Project Name is created to access a project further in future. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Purpose is to see projects in the directory.

Table : Description of FR-13

|  |  |
| --- | --- |
| **Identifier** | FR-13 |
| **Title** | Accessing previous projects |
| **Requirement** | List of previous projects will be visible to the user. |
| **Source** | User |
| **Rationale** | Purpose is to see projects in the directory. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

The purpose is to facilitate user to resume his/her work from where left. Or to go through a project that was previously created.

Table : Description of FR-14

|  |  |
| --- | --- |
| **Identifier** | FR-14 |
| **Title** | Selecting a project |
| **Requirement** | The user shall be able to open any previous projects in the list. |
| **Source** | User |
| **Rationale** | The purpose is to facilitate user to resume his/her work from where left. Or to go through a project that was previously created. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-12 |
| **Priority** | Medium |

### Functional Requirement for Modes

UI Mode is the core purpose of our project. Selecting this mode will enable users to work on Drag and Drop Mode and build react applications.

Table : Description of FR-15

|  |  |
| --- | --- |
| **Identifier** | FR-15 |
| **Title** | Selecting UI mode |
| **Requirement** | The user shall click on UI mode in the top header of the software. |
| **Source** | User |
| **Rationale** | UI Mode is the core purpose of our project. Selecting this mode will enable users to work on Drag and Drop Mode and build react applications. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-14 |
| **Priority** | High |

UI Mode is the core purpose of our project. Selecting this mode will enable users to work on Drag and Drop Mode and build react applications. There will be different ideas and categories to select from to build an application.

Table : Description of FR-16

|  |  |
| --- | --- |
| **Identifier** | FR-15 |
| **Title** | Creating Applications in UI mode |
| **Requirement** | The user shall click on UI mode in the top header of the software, and create apps with listed templates and components. |
| **Source** | User |
| **Rationale** | UI Mode is the core purpose of our project. Selecting this mode will enable users to work on Drag and Drop Mode and build react applications. There will be different ideas and categories to select from to build an application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-15 |
| **Priority** | High |

Developing Apps with debugging. For users who prefer debugging.

Table : Description of FR-17

|  |  |
| --- | --- |
| **Identifier** | FR-17 |
| **Title** | Selecting coding mode |
| **Requirement** | The user shall click on code mode in the top header of the software. |
| **Source** | User |
| **Rationale** | Developing Apps with debugging. For users who prefer debugging. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-14 |
| **Priority** | High |

Selecting this mode will enable users to work on coding mode and build react applications, with debugging. Here the user uses the terminal and an IDE to write and run his/her code.

Table : Description of FR-18

|  |  |
| --- | --- |
| **Identifier** | FR-17 |
| **Title** | Creating Applications Coding Mode. |
| **Requirement** | The user shall write a code to generate his application. |
| **Source** | User |
| **Rationale** | Selecting this mode will enable users to work on coding mode and build react applications, with debugging. Here the user uses the terminal and an IDE to write and run his/her code. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-17 |
| **Priority** | Medium |

### Functional Requirement for preserving and updating work

Purpose is to save the user from clicking ctrl + s every time.

Table : Description of FR-19

|  |  |
| --- | --- |
| **Identifier** | FR-19 |
| **Title** | Preserving work |
| **Requirement** | The user will have to do nothing to save his/her work |
| **Source** | User |
| **Rationale** | This purpose is to save the user from clicking ctrl + s every time. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

This may be very useful in times of unexpected system failures. As the file will keep updating after every 10 seconds.

Table : Description of FR-20

|  |  |
| --- | --- |
| **Identifier** | FR-20 |
| **Title** | Updating file. |
| **Requirement** | The user will have to do nothing to update his/her work |
| **Source** | User |
| **Rationale** | This may be very useful in times of unexpected system failures. As the file will keep updating after every 10 seconds. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

### Functional Requirement for selecting from store

Templates are divided into categories; the user can opt for his/her specific category.

Table : Description of FR-21

|  |  |
| --- | --- |
| **Identifier** | FR-21 |
| **Title** | Selecting A template |
| **Requirement** | The user shall be able to select a template |
| **Source** | User |
| **Rationale** | Templates are divided into categories; the user can opt for his/her specific category. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Installing the template will include that template in his modules.

Table : Description of FR-22

|  |  |
| --- | --- |
| **Identifier** | FR-22 |
| **Title** | Installing a Template |
| **Requirement** | The user can install the selected template |
| **Source** | User |
| **Rationale** | Installing the template will include that template in his modules. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-21 |
| **Priority** | High |

Templates can be used, to get a good, elegant, and user-friendly pre-design to the developing application.

Table : Description of FR-23

|  |  |
| --- | --- |
| **Identifier** | FR-23 |
| **Title** | Using A Template |
| **Requirement** | The user will use the template in the application. |
| **Source** | User |
| **Rationale** | Templates can be used, to get a good, elegant and user-friendly pre-design to the developing application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-21, FR-22 |
| **Priority** | High |

There are 100 + components for the user to select from.

Table : Description of FR-24

|  |  |
| --- | --- |
| **Identifier** | FR-24 |
| **Title** | Selecting A Component |
| **Requirement** | The user shall be able to select a Component |
| **Source** | User |
| **Rationale** | There are 100 + components for the user to select from. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Downloading the component will provide accessibility to use that component in the project.

Table : Description of FR-25

|  |  |
| --- | --- |
| **Identifier** | FR-25 |
| **Title** | Downloading a Component |
| **Requirement** | The user can install the selected component |
| **Source** | User |
| **Rationale** | Downloading the component will provide accessibility to use that component in the project. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-24 |
| **Priority** | High |

Components which are downloaded could be used to enhance the look and feel of the application.

Table : Description of FR-26

|  |  |
| --- | --- |
| **Identifier** | FR-26 |
| **Title** | Using A Component |
| **Requirement** | The user will use the component in the application. |
| **Source** | User |
| **Rationale** | Components which are downloaded could be used to enhance the look and feel of the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-24, FR-25 |
| **Priority** | High |

### Functional Requirement for screen building

The user shall be able to add a new screen to the project, by clicking add button.

Table : Description of FR-27

|  |  |
| --- | --- |
| **Identifier** | FR-27 |
| **Title** | Add a screen in UI Mode |
| **Requirement** | The user shall be able to add a new screen to the project, by clicking add button. |
| **Source** | User |
| **Rationale** | Adding screens to start the build. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user shall click the delete button to delete a screen

Table : Description of FR-28

|  |  |
| --- | --- |
| **Identifier** | FR-28 |
| **Title** | Delete a screen in UI Mode |
| **Requirement** | The user shall click the delete button to delete a screen |
| **Source** | User |
| **Rationale** | The user can delete any previously created screen if he wants to delete it. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-27 |
| **Priority** | Medium |

The user shall be able to right click and rename the screen

Table : Description of FR-29

|  |  |
| --- | --- |
| **Identifier** | FR-29 |
| **Title** | Rename a screen in UI Mode |
| **Requirement** | The user shall be able to right click and rename the screen |
| **Source** | User |
| **Rationale** | Enabling user to rename a screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-27 |
| **Priority** | Medium |

The user shall be able to add a new screen to the project, by clicking on the left panel.

Table : Description of FR-30

|  |  |
| --- | --- |
| **Identifier** | FR-30 |
| **Title** | Add a screen in Code Mode |
| **Requirement** | The user shall be able to add a new screen to the project, by clicking on the left panel. |
| **Source** | User |
| **Rationale** | Adding screens to start the build. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user shall right click the selected file in the left panel and delete it.

Table : Description of FR-31

|  |  |
| --- | --- |
| **Identifier** | FR-31 |
| **Title** | Delete a screen in Code Mode |
| **Requirement** | The user shall right click the selected file in the left panel and delete it. |
| **Source** | User |
| **Rationale** | The user can delete any previously created screen if he wants to delete it. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-30 |
| **Priority** | Medium |

The user shall right click the selected file in the left panel and rename it.

Table : Description of FR-32

|  |  |
| --- | --- |
| **Identifier** | FR-32 |
| **Title** | Rename a screen in Code Mode |
| **Requirement** | The user shall right click the selected file in the left panel and rename it. |
| **Source** | User |
| **Rationale** | Enabling user to rename a screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-30 |
| **Priority** | Medium |

### Functional Requirement for navigation

The user will select a type of navigation from Tab Navigator, Stack Navigator, Drawer Navigator.

Table : Description of FR-33

|  |  |
| --- | --- |
| **Identifier** | FR-33 |
| **Title** | Selecting a navigation in UI mode |
| **Requirement** | The user will select a type of navigation from Tab Navigator, Stack Navigator, Drawer Navigator. |
| **Source** | User |
| **Rationale** | Purpose is to choose a method of navigation/linking from screen to screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen.

Table : Description of FR-34

|  |  |
| --- | --- |
| **Identifier** | FR-34 |
| **Title** | Adding stack Navigator in UI mode |
| **Requirement** | The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen. |
| **Source** | User |
| **Rationale** | Its purpose is to create stacks of screens. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-33 |
| **Priority** | Medium |

The user shall be able to use the props of the navigator listed at the side of the working panel.

Table : Description of FR-35

|  |  |
| --- | --- |
| **Identifier** | FR-35 |
| **Title** | Using props for the stack navigator in UI Mode |
| **Requirement** | The user shall be able to use the props of the navigator listed at the side of the working panel. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-33 |
| **Priority** | Medium |

The user shall be able to style the navigator.

Table : Description of FR-36

|  |  |
| --- | --- |
| **Identifier** | FR-36 |
| **Title** | Styling the stack navigator in UI Mode. |
| **Requirement** | The user shall be able to style the navigator. |
| **Source** | User |
| **Rationale** | For beautifying the navigator. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-33 |
| **Priority** | Low |

The user shall be able to delete an already created navigator.

Table : Description of FR-37

|  |  |
| --- | --- |
| **Identifier** | FR-37 |
| **Title** | Deleting the stack navigator in UI Mode. |
| **Requirement** | The user shall be able to delete an already created navigator. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator with one click |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-33 |
| **Priority** | High |

The user shall be able to add a navigator by coding it.

Table : Description of FR-38

|  |  |
| --- | --- |
| **Identifier** | FR-38 |
| **Title** | Adding stack Navigator in code mode |
| **Requirement** | The user shall be able to add a navigator by coding it. |
| **Source** | User |
| **Rationale** | Its purpose is to create stacks of screens. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Low |

The user shall be able to use the props of the navigator in his code.

Table : Description of FR-39

|  |  |
| --- | --- |
| **Identifier** | FR-39 |
| **Title** | Using props for the stack navigator in code Mode |
| **Requirement** | The user shall be able to use the props of the navigator in his code. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-38 |
| **Priority** | Medium |

The user shall be able to do styling of the navigator.

Table : Description of FR-40

|  |  |
| --- | --- |
| **Identifier** | FR-40 |
| **Title** | Styling the stack navigator in code Mode. |
| **Requirement** | The user shall be able to do styling of the navigator. |
| **Source** | User |
| **Rationale** | Styling can be done either inline or in a stylesheet, as per users choice. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-38 |
| **Priority** | Medium |

The user shall be able to delete the navigator by erasing the code.

Table : Description of FR-41

|  |  |
| --- | --- |
| **Identifier** | FR-41 |
| **Title** | Deleting the stack navigator in code Mode. |
| **Requirement** | The user shall be able to delete the navigator by erasing the code. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen.

Table : Description of FR-42

|  |  |
| --- | --- |
| **Identifier** | FR-42 |
| **Title** | Adding tab Navigator in UI mode |
| **Requirement** | The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen. |
| **Source** | User |
| **Rationale** | Its purpose is to create a tab navigator. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | NA |
| **Priority** | Medium |

Props needed to be altered according to the properties needed in the application.

Table : Description of FR-43

|  |  |
| --- | --- |
| **Identifier** | FR-43 |
| **Title** | Using props for the tab navigator in UI Mode |
| **Requirement** | The user shall be able to to use the props of the navigator listed at the side of the working panel. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-42 |
| **Priority** | Medium |

For beautifying the navigator.

Table : Description of FR-44

|  |  |
| --- | --- |
| **Identifier** | FR-44 |
| **Title** | Styling the tab navigator in UI Mode. |
| **Requirement** | The user shall be able to style the navigator. |
| **Source** | User |
| **Rationale** | For beautifying the navigator. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-42 |
| **Priority** | Low |

The user shall be able to delete an already created navigator.

Table : Description of FR-45

|  |  |
| --- | --- |
| **Identifier** | FR-45 |
| **Title** | Deleting the tab navigator in UI Mode. |
| **Requirement** | The user shall be able to delete an already created navigator. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator with one click |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-42 |
| **Priority** | High |

Its purpose is to create tab navigator either at the bottom or top of the screen.

Table : Description of FR-46

|  |  |
| --- | --- |
| **Identifier** | FR-46 |
| **Title** | Adding tab Navigator in code mode |
| **Requirement** | The user shall be able to add a navigator by coding it. |
| **Source** | User |
| **Rationale** | Its purpose is to create tab navigator either at the bottom or top of the screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Low |

Props needed to be altered according to the properties needed in the application.

Table : Description of FR-47

|  |  |
| --- | --- |
| **Identifier** | FR-47 |
| **Title** | Using props for the tab navigator in code Mode |
| **Requirement** | The user shall be able to use the props of the navigator in his code. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR- 46 |
| **Priority** | Medium |

Styling can be done either inline or in a stylesheet, as per users’ choice.

Table : Description of FR-48

|  |  |
| --- | --- |
| **Identifier** | FR-48 |
| **Title** | Styling the tab navigator in code Mode. |
| **Requirement** | The user shall be able to do styling of the navigator. |
| **Source** | User |
| **Rationale** | Styling can be done either inline or in a stylesheet, as per users’ choice. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-46 |
| **Priority** | Medium |

Enabling the user to delete a navigator

Table : Description of FR-49

|  |  |
| --- | --- |
| **Identifier** | FR-49 |
| **Title** | Deleting the tab navigator in code Mode. |
| **Requirement** | The user shall be able to delete the navigator by erasing the code. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-46 |
| **Priority** | Medium |

The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen.

Table : Description of FR-50

|  |  |
| --- | --- |
| **Identifier** | FR-50 |
| **Title** | Adding drawer Navigator in UI mode |
| **Requirement** | The user shall be able to add the navigator by just dragging it from the panel and dropping it on the desired position at the screen. |
| **Source** | User |
| **Rationale** | Its purpose is to create a drawer navigator on either end of the screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | NA |
| **Priority** | Medium |

Props needed to be altered according to the properties needed in the application.

Table : Description of FR-51

|  |  |
| --- | --- |
| **Identifier** | FR-51 |
| **Title** | Using props for the drawer navigator in UI Mode |
| **Requirement** | The user shall be able to use the props of the navigator listed at the side of the working panel. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-50 |
| **Priority** | Medium |

For beautifying the navigator.

Table : Description of FR-52

|  |  |
| --- | --- |
| **Identifier** | FR-52 |
| **Title** | Styling the drawer navigator in UI Mode. |
| **Requirement** | The user shall be able to style the navigator. |
| **Source** | User |
| **Rationale** | For beautifying the navigator. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-50 |
| **Priority** | Low |

Enabling the user to delete a navigator with one click

Table : Description of FR-53

|  |  |
| --- | --- |
| **Identifier** | FR-53 |
| **Title** | Deleting the drawer navigator in UI Mode. |
| **Requirement** | The user shall be able to delete an already created navigator. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator with one click |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-53 |
| **Priority** | High |

Its purpose is to create drawer navigator either at the bottom or top of the screen.

Table : Description of FR-54

|  |  |
| --- | --- |
| **Identifier** | FR-54 |
| **Title** | Adding drawer Navigator in code mode |
| **Requirement** | The user shall be able to add a navigator by coding it. |
| **Source** | User |
| **Rationale** | Its purpose is to create drawer navigator either at the bottom or top of the screen. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Low |

Props needed to be altered according to the properties needed in the application.

Table : Description of FR-55

|  |  |
| --- | --- |
| **Identifier** | FR-55 |
| **Title** | Using props for the drawer navigator in code Mode |
| **Requirement** | The user shall be able to use the props of the navigator in his code. |
| **Source** | User |
| **Rationale** | Props needed to be altered according to the properties needed in the application. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR- 54 |
| **Priority** | Medium |

Styling can be done either inline or in a stylesheet, as per users’ choice.

Table : Description of FR-56

|  |  |
| --- | --- |
| **Identifier** | FR-56 |
| **Title** | Styling the drawer navigator in code Mode. |
| **Requirement** | The user shall be able to do styling of the navigator. |
| **Source** | User |
| **Rationale** | Styling can be done either inline or in a stylesheet, as per users’ choice. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-54 |
| **Priority** | Medium |

Enabling the user to delete a navigator

Table : Description of FR-57

|  |  |
| --- | --- |
| **Identifier** | FR-57 |
| **Title** | Deleting the drawer navigator in code Mode. |
| **Requirement** | The user shall be able to delete the navigator by erasing the code. |
| **Source** | User |
| **Rationale** | Enabling the user to delete a navigator |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-54 |
| **Priority** | Medium |

### Functional Requirement for terminal commands

Enabling the user to write a command on the terminal.

Table : Description of FR-58

|  |  |
| --- | --- |
| **Identifier** | FR-58 |
| **Title** | Writing a command |
| **Requirement** | The user shall be able to write any relevant command on the terminal |
| **Source** | User |
| **Rationale** | Enabling the user to write a command on the terminal. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Its purpose is to execute the command to support the system.

Table : Description of FR-59

|  |  |
| --- | --- |
| **Identifier** | FR-59 |
| **Title** | Executing a command |
| **Requirement** | The user shall be able to run any relevant command from the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to execute the command to support the system. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-58 |
| **Priority** | Medium |

Its purpose is to suggest users with appropriate commands.

Table : Description of FR-60

|  |  |
| --- | --- |
| **Identifier** | FR-60 |
| **Title** | Searching for command |
| **Requirement** | The user shall be able to search a relevant command by writing key words on the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to suggest users with appropriate commands. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Enabling the user to write a command on the terminal.

Table : Description of FR-61

|  |  |
| --- | --- |
| **Identifier** | FR-61 |
| **Title** | Writing a windows command |
| **Requirement** | The user shall be able to write any relevant command on the terminal |
| **Source** | User |
| **Rationale** | Enabling the user to write a command on the terminal. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Its purpose is to execute the command to support the system.

Table : Description of FR-62

|  |  |
| --- | --- |
| **Identifier** | FR-62 |
| **Title** | Executing a windows command |
| **Requirement** | The user shall be able to run any relevant command from the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to execute the command to support the system. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-58 |
| **Priority** | Medium |

Its purpose is to suggest users with appropriate commands.

Table : Description of FR-63

|  |  |
| --- | --- |
| **Identifier** | FR-63 |
| **Title** | Searching for windows command |
| **Requirement** | The user shall be able to search a relevant command by writing key words on the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to suggest users with appropriate commands. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Enabling the user to write a command on the terminal.

Table : Description of FR-64

|  |  |
| --- | --- |
| **Identifier** | FR-64 |
| **Title** | Writing a git command |
| **Requirement** | The user shall be able to write any relevant command on the terminal |
| **Source** | User |
| **Rationale** | Enabling the user to write a command on the terminal. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Its purpose is to execute the command to support the system.

Table : Description of FR-65

|  |  |
| --- | --- |
| **Identifier** | FR-65 |
| **Title** | Executing a git command |
| **Requirement** | The user shall be able to run any relevant command from the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to execute the command to support the system. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-64 |
| **Priority** | Medium |

Its purpose is to suggest users with appropriate commands.

Table : Description of FR-66

|  |  |
| --- | --- |
| **Identifier** | FR-66 |
| **Title** | Searching for git command |
| **Requirement** | The user shall be able to search a relevant command by writing key words on the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to suggest users with appropriate commands. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

Enabling the user to write a command on the terminal.

Table : Description of FR-67

|  |  |
| --- | --- |
| **Identifier** | FR-67 |
| **Title** | Writing a npm command |
| **Requirement** | The user shall be able to write any relevant command on the terminal |
| **Source** | User |
| **Rationale** | Enabling the user to write a command on the terminal. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Its purpose is to execute the command to support the system.

Table : Description of FR-68

|  |  |
| --- | --- |
| **Identifier** | FR-68 |
| **Title** | Executing a npm command |
| **Requirement** | The user shall be able to run any relevant command from the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to execute the command to support the system. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-67 |
| **Priority** | Medium |

Its purpose is to suggest users with appropriate commands.

Table : Description of FR-69

|  |  |
| --- | --- |
| **Identifier** | FR-69 |
| **Title** | Searching for npm command |
| **Requirement** | The user shall be able to search a relevant command by writing key words on the terminal. |
| **Source** | User |
| **Rationale** | Its purpose is to suggest users with appropriate commands. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

If a command is not found, or a command is mis spelled the terminal throws an error to inform the user.

Table : Description of FR-70

|  |  |
| --- | --- |
| **Identifier** | FR-70 |
| **Title** | Command Exception |
| **Requirement** | The user shall execute a command on the terminal. |
| **Source** | User |
| **Rationale** | If a command is not found, or a command is mis spelled the terminal throws an error to inform the user. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | FR-67 |
| **Priority** | High |

### Functional Requirement for version controlling

Enabling the user to work on version control system environment.

Table : Description of FR-91

|  |  |
| --- | --- |
| **Identifier** | FR-91 |
| **Title** | Cloning a project |
| **Requirement** | The user shall be able to clone a project from git hub. |
| **Source** | User |
| **Rationale** | Enabling the user to work on version control system environment. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Enabling the user to work on version control system environment, adding their work on git hub.

Table : Description of FR-92

|  |  |
| --- | --- |
| **Identifier** | FR-92 |
| **Title** | Add a code to git |
| **Requirement** | The user shall be able to add your work on a project in git hub |
| **Source** | User |
| **Rationale** | Enabling the user to work on version control system environment, adding their work on git hub. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Enabling the user to work on version control system environment.

Table : Description of FR-93

|  |  |
| --- | --- |
| **Identifier** | FR-93 |
| **Title** | Commit on git |
| **Requirement** | The user shall be able to commit the changes on git hub. |
| **Source** | User |
| **Rationale** | Enabling the user to work on version control system environment. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Enabling the user to work on version control system environment, pushing on git hub.

Table : Description of FR-94

|  |  |
| --- | --- |
| **Identifier** | FR-94 |
| **Title** | Push on git |
| **Requirement** | The user shall be able to push your work on a project in git hub |
| **Source** | User |
| **Rationale** | Enabling the user to work on version control system environment, pushing on git hub. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Enabling the user to work on version control system environment, pulling from git hub.

Table : Description of FR-95

|  |  |
| --- | --- |
| **Identifier** | FR-95 |
| **Title** | Pull on git |
| **Requirement** | The user shall be able to pull your work from a project in git hub |
| **Source** | User |
| **Rationale** | Enabling the user to work on version control system environment, pulling from git hub. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | High |

Branching on git helps to maintain your work in a sperate branch which can be merged in the future.

Table : Description of FR-96

|  |  |
| --- | --- |
| **Identifier** | FR-96 |
| **Title** | Branch on git |
| **Requirement** | The user shall be able to create a branch and commit his/her changes on it. |
| **Source** | User |
| **Rationale** | Branching on git helps to maintain your work in a sperate branch which can be merged in the future. |
| **Business Rule (if required)** | N/A |
| **Dependencies** | N/A |
| **Priority** | Medium |

## Non-Functional Requirements

My system will be available 24/7. Anyone with a pc can access my software from any place around the globe.

### Reliability

* Our system will have MTBF of 30s.

### Usability

* End-User shall be able to learn the system functionality very quickly.
* System shall provide help about new features.

### Performance

* System shall load web page within 5 seconds on 20Mbps or 3G or faster internet connection.
* Average response time for each query shall be less than 5 seconds.
* System shall load templates within 10 min.
* System shall load components within 1 min.

### Security

* System shall restrict one user to view information of another.
* System shall restrict any third-party tools to get cookies from our software.
* System shall verify each user account via email, each email may be registered once.

## External Interface Requirements

RCC will communicate efficiently with the user and all software elements. There are no external hardware systems in RCC.

### User Interfaces Requirements

Efficient windows management- as the user doesn’t need to move the mouse extensively.

* The very generic design is followed in the title bar. Close, Minimize, Maximize/Restore buttons.
* The application window represents the name of the software.
* Each panel displays its descriptive title. (Multiple items are separated with a -)
* The Sw is fully responsive at all screen sizes. But a limit is set for minimum window size.
* The window’s contents changes in “real time” as the user drags the scroller.
* RCC ensures persistency, the window remains in the same state as last accessed.

The aesthetics of RCC:

* Simple and realistic icons are used, which users are already familiar to.
* Visible, clear, and easy to read font styles and sizes are incorporated. The font body is left aligned.
* The primary color used is grey, denoting maturity and composure. Incorporated with brown, white, and blue.
* resolution 1080 x 720 till 4K.

The interaction of the system is made as smooth as possible:

* If a user presses down the mouse button while the pointer is over an onscreen button, thereby putting the button in a selected state, and then moves the pointer off the button before releasing the mouse button, the onscreen button is not clicked. If the user presses an onscreen button and rolls over another button before releasing the mouse, neither button is clicked.
* Double click comes in handy at certain places but it is not the only option as users do not use it most commonly.
* The uses of dragging include selecting components and setting it on the desired coordinates on the canvas. RCC restricts an object from being moved past certain boundaries, such as the edge of the canvas. If the user drags an object and releases the mouse button outside the boundary, the object stays in the original location. If the user drags the item out of the boundary and then back in before releasing the mouse button, the object moves to the new location.
* Hot keys have been added to get the user in ease.
* Dark background is chosen so that it is useful to visually impaired users.
* The two top bars remain at every screen as they contain standard buttons, navigation to all screens, change of modes to UI and coding and etc.

Standard buttons, functions, or navigation links that will appear on every screen, such as a help button.

Message displaying convention is easy to read, understand. It is written in the simplest language and also provides an alternate solution to the user.

Document the user interface design details, such as specific dialog box layouts, in a separate user interface specification, not in the SRS. Including screen mock-ups in the SRS to communicate another view of the requirements is helpful but make it clear that the mock-ups are not the committed screen designs. If the SRS is specifying an enhancement to an existing system, it sometimes makes sense to include screen displays exactly as they are to be implemented. The developers are already constrained by the current reality of the existing system, so it's possible to know up front just what the modified, and perhaps the new, screens should look like.

### Software interfaces

RCC relates to no other software interfaces currently.

### Hardware interfaces

Not Available.

### Communications interfaces

CI-1: When there is sign up to the system a confirmation email will be sent to the relative email address.

# Design and Architecture

The following parts includes Software Design Description (SDD) report of RCC. The architecture of RCC, its flow of work, sequence of data flow and the activities performed.



## Architectural Design

Users will need to install the application. After the installation is finished, users can create a new project, create a whole application from scratch or use our free or paid components or templates. Once the application is made, a user can also generate its build to release on google play store or apple app store. This whole app development process does not need an internet connection except when a user needs to install some dependencies from the terminal through npm or download our templates through our own package manager.

Diagram

Description automatically generated

Figure : Line-box

## Design Models

The design models applicable for RCC are as follows:

### Activity Diagram

The diagram below depicts the entire flow of creating a project. Entering the project information and then saving it in the system.

Diagram

Description automatically generated

Figure Activity Diagram for Initializing Project.

The diagram below depicts how the user will traverse a desired project from a list of created projects.

Diagram

Description automatically generated

Figure Activity Diagram for Open an existing Project.

The diagram below depicts how the user will shift between modes, these tasks are working parallel.

Diagram

Description automatically generated

Figure Activity Diagram for Toggle Modes.

The diagram below depicts the auto-save feature of the software.

Diagram

Description automatically generated

Figure Activity Diagram for Save Projects.

The diagram below depicts the flow of the package manager module. How the user will select a component and template and use it in the application.

Diagram

Description automatically generated

Figure 9 Activity Diagram for Access Store of templates.

The diagram below depicts the flow of the package manager module. How the user will select a component and template and use it in the application.

Diagram

Description automatically generated

Figure Activity Diagram for Access Store of components.

The diagram below depicts the management of components in the project.

Diagram

Description automatically generated

Figure Activity Diagram for Manage Component.

The diagram below depicts the flow of embedding third-party tools like firebase and agora.

Diagram

Description automatically generated

Figure Activity Diagram for Integrate Third Party Tools.

The diagram below depicts the flow of adding navigation.

Diagram, application

Description automatically generated

Figure Activity Diagram for Managing Navigation.

The diagram below depicts the flow of stack navigation its styling.

Diagram

Description automatically generated

Figure Activity Diagram for Stack Navigation.

The diagram below depicts the flow of tab navigation its styling.

Diagram

Description automatically generated

Figure Activity Diagram for Tab Navigation.

The diagram below depicts the flow of how the user will build screens, drag components to the screen and adding the related styling, props to the respective components.

A picture containing text

Description automatically generated

Figure Activity Diagram for Screen Building.

The diagram below depicts the flow of how the user will interact with the terminal typing and executing commands related to windows, MAC, git and NPM.

Diagram

Description automatically generated

Figure Activity Diagram for Interacting with the Terminal.

The diagram below depicts the flow of how the user will interact with the terminal executing all windows commands.

Diagram

Description automatically generated

Figure Activity Diagram for Running Windows Command.

The diagram below depicts the flow of how the user will interact with the terminal executing all git commands

Diagram

Description automatically generated

Figure Activity Diagram for Running Git Command.

The diagram below depicts the flow of how the user will interact with the terminal executing all NPM commands

Diagram

Description automatically generated

Figure Activity Diagram for Running NPM Command.

The diagram below depicts the flow of how the user will perform version control. Selecting add, commit, and push.

Diagram

Description automatically generated

Figure Activity Diagram for Performing Version Control.

### Data flow diagram

Below is the data flow diagram of level 0 depicting the main functionalities of React Code Catalyzer. The software’s interaction with the terminal. Building of screens and the display of application on the emulator and PD.

Diagram

Description automatically generated

Figure Data Flow Diagram of Level 0.

Below is the data flow diagram of level 1 depicting how projects will be managed, how templates and components will be used, how commands will be run from the built-in terminal, how user will perform version control system to add and remove changes and how customized template will be generated in an application.

Diagram

Description automatically generated

Figure Data Flow Diagram of Level 1

Below is the data flow diagram of level 2(part 1) depicting the use of terminal, app development, managing projects and automated template generation with added details.

Diagram

Description automatically generated

Figure Data Flow Diagram of Level 2a.

Below is the data flow diagram of level 2(part 2) depicting the managing of store(components/templates) and GitHub integration to perform version control system(clone, pull, push, add, commit).

Diagram, schematic

Description automatically generated

Figure Data Flow Diagram of Level 2b.

### Behavioral Diagram

As the approach used for building RCC is procedural. We have denoted the flow of the system with a control flow diagram.

#### Control Flow Diagram

The control flow diagram below depicts the interaction of the user with the terminal, various servers to build application and the webstore.

For the terminal the user will first send a new command to the command line simultaneously the command will also be sent to the invoker. The command is then stored in the invoker, it is sent to the command line to get executed. At the end the receiver gets what must be done.

The user will send a request to the system to initiate an application (the request will be npm init) the system will respond by creating the application.

To perform version, control the user will send request to the system. The system will further the request to git servers and get back the response.

In the similar manner to generate a customized template the user will send request to the system, the request will be forwarded to the server, related data from the server will be extracted, template will be scrapped and created.

The interaction of user with the webstore to download and add in components and templated to the application. The request is first sent to the web server, it is then forwarded to the application server from there it goes to the database server which in our case is the firebase, the result set is returned, the page is dynamically generated, and the user gets back the response.

Diagram

Description automatically generated

Figure 26 Control flow diagram depicting working of RCC

## Data Design

Terminal: {

required: ["userName", "computer", "currentPath", "command"],

properties: {

userName: {

Type: "string",

description: "Name of the user",

}

computer: {

Type: "string",

description: "Name of the computer",

}

currentPath: {

Type: "string",

description: "The current directory",

}

command: {

Type: "string",

}

}

}

The components/templates and changes made in the application by the user are stored in the local machine of the user. Majorly there is processing on our system.

### Data Dictionary

|  |
| --- |
| **Function Name** |
| Terminal() |
| Navbar() |
| createNewFile() |
| addNewExplorerTabInFilesContainer() |
| openCodeSlate() |
| loadView() |
| Offline() |
| showAuthorDetails() |
| widgetLogic() |
| showWidget() |
| initWebView() |
| loadStart() |
| loadStop() |
| Forward() |
| Backward() |
| hideWidget() |
| editorMode() |
| onChange() |
| UIMode() |

Terminal() fields:

|  |  |
| --- | --- |
| **Fields** | **Data Type** |
| userName | VARCHAR |
| computer | VARCHAR |
| currentPath | VARCHAR |
| command | VARCHAR |

createNewFile() fields:

|  |  |
| --- | --- |
| **Fields** | **Data Type** |
| projectName | VARCHAR |
| directory | VARCHAR |

## Human Interface Design

RCC is a self-explanatory tool which is easy to use and do not require any manual. RCC is constructed keeping in mind user experiences. Following a consistent design so that the user can easily acclimatize.

### Screen Images

This is how the user will perceive the drag N drop mode UI of RCC. It is a very simple and neat design. Selecting components from the left panel and applying styles to the respective component from the right panel.

Shape

Description automatically generated with medium confidence

Figure Drag N Drop mode from user's perspective

This is how the user will perceive the drag N drop mode UI of RCC. The design shows consistency as it is like well-known IDEs (VSCode).

Graphical user interface, text

Description automatically generated

Figure Coding mode from user's perspective

## Screen Objects and Actions

Components will be clicked and dropped onto the working board. Their coordinates will be changed by dragging the cursor on the canvas.

As soon as a component is selected, it’s props and styles will be enabled, and the user can adjust the styling from there.

The buttons on the top right bar allow user(s) to perform version controlling.

# Implementation

Implementation details of RCC are listed below, from how the project is initialized till generating a release apk of the project. All core module functionalities in pseudocode form is here.



## Algorithm

The table below shows the pseudocode for creating a react native application using RCC. Building and adding screens to the project. Performing version control with GUI. Not needing to write and execute any command. Lastly, users’ interaction with the terminal is also added.

Table Core algorithm implementation table.

|  |
| --- |
| **Algorithm 1 Creating a Project** |
| **Input:** projectName and template |
| **Output:** Project created in dir |
| 1: dir ← ‘C:/React Code Catalyzer’  2: if (not exists(dir))  3: create(dir)  4: output ← execute(“cd ” + dir + “ && expo init ” + projectName.split(“ ”).join(“”) + “ --template ” + template)  5: if (template == ‘blank’)  6: execute(“yarn add @react-navigation/native”)  7: execute(“expo install react-native-screens react-native-safe-area-context”)  8: execute(“yarn add @react-navigation/native-stack”)  9: end if  10: end if  11: else  12: output ← execute(“cd ” + dir + “ && expo init ” + projectName.split(“ ”).join(“”) + “ --template ” + template)  13: if (template == ‘blank’)  14: execute(“yarn add @react-navigation/native”)  15: execute(“expo install react-native-screens react-native-safe-area-context”)  16: execute(“yarn add @react-navigation/native-stack”)  17: end if  18: end else |
| **Algorithm 2 Screen Building** |
| **Input:** screenName |
| **Output:** Boilerplate of screen built |
| 1: openedProject ← ‘src/config/OpenedProject.json’  2: content ← readFile(openedProject)  3: if (current number of screens == 0)  4: makeDirectory(content.location + ‘/src’)  5: makeDirectory(content.location + ‘/src/screens’)  6: makeDirectory(content.location + ‘/src/screens/’ + screenName)  7: screenInitialization ←   |  | | --- | | `import React from 'react' | |  | import { View, Text } from 'react-native' | |  | import styles from './styles' | |  |  | |  | const ${projectName} = () => { | |  | return ( | |  | <View style = {{flex: 1, alignItems: 'center', justifyContent: 'center', backgroundColor: 'white'}}> | |  | <Text>${projectName}</Text> | |  | </View> | |  | ) | |  | } | |  |  | |  | export default ${projectName}` |   8: writeFile(content.location + ‘/src/screens/’ + screenName + screenName + “.js”, screenInitialization)  9: end if |
| **Algorithm 3 Git Integration** |
| **Input:** message |
| **Output:** Required git command executed |
| 1: **//commit function**  2: execute(“git add . && git commit -m ” + message)  3: **//branch function**  4: execute(“git checkout -b ” + message)  5: **//clone function**  6: execute(“echo "# Test1" >> README.md && git init && git add . && git commit -m "first commit" && git branch -M main && git remote add origin ” + {message} + “ && git push -u origin main”)  7: **//push function**  8: execute(“git push”) |
| **Algorithm 4 Terminal** |
| **Input:** command |
| **Output:** Command executed in Terminal |
| 1: if(<enter> is pressed)  2: //prevent default behaviour of <enter>  3: output ← execute(command)  4: display(output) |
| **Algorithm 5 Run Project** |
| **Input:** command |
| **Output:** Run Project window opens in browser |
| 1: startOutput ← execute(“expo start”)  2: if(command is executed successfully)  3: open\_window(“http://localhost:19002”)  4: else  5: print(error) |

## User Interface

Details about user interface with descriptions will be presented in this section.

Graphical user interface

Description automatically generated

Figure : UI for project creation.

* The first step for the user is to initiate a project, just pressing the CREATE button (npm init).
* By selected the blank option the user creates a blank project, by selecting tabs you can create a project having bottom tabs.

Graphical user interface, application

Description automatically generated

Figure : UI for adding screens

* This option enables user to continue adding multiple screens.

Graphical user interface, application

Description automatically generated

Figure : UI for drag N drop mode.

* This UI depicts the drag N drop mode of RCC.
* On the left panel the set of components are present while on the right panel there is styling and props for the selected component.
* The top bar has version control with just a click.

A screenshot of a computer

Description automatically generated with medium confidence

Figure : UI for coding mode.

* This is the coding UI with an intelli sense-controlled IDE.
* The bottom contains a fully functional terminal.
* The top bar has version control with just a click.

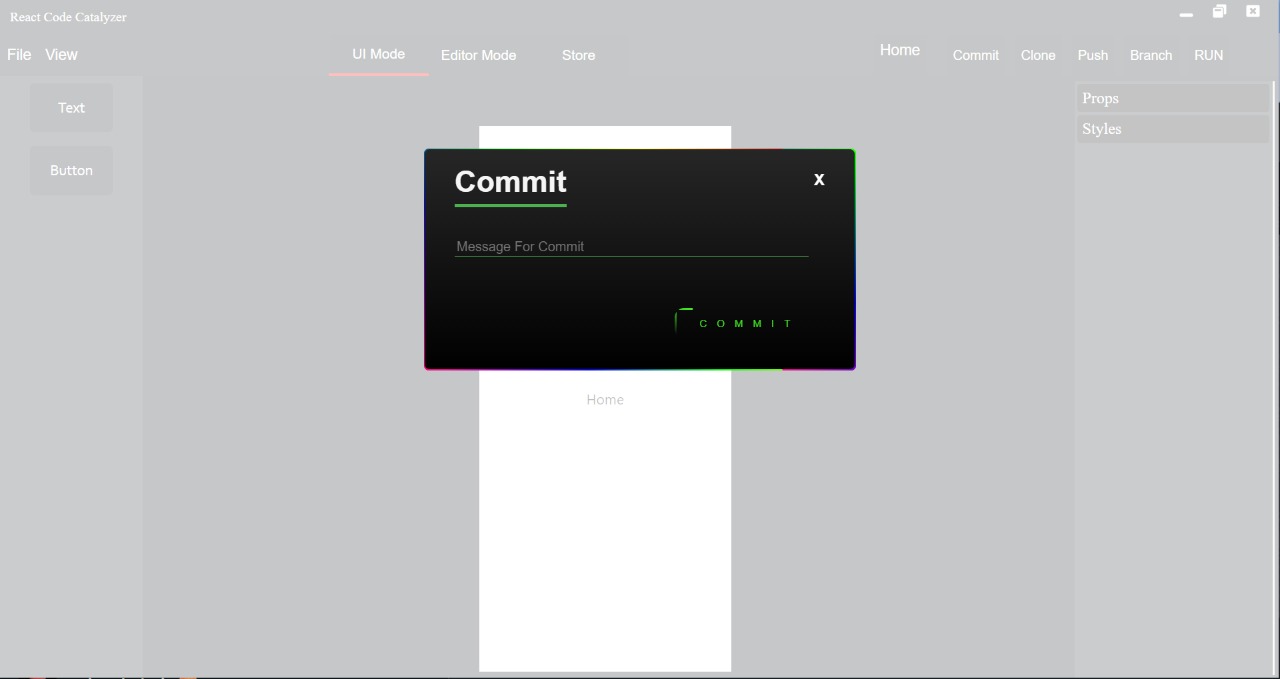


Figure : UI for git commit command

* This commit modal allows user to type related message and commit changes to git with a click.

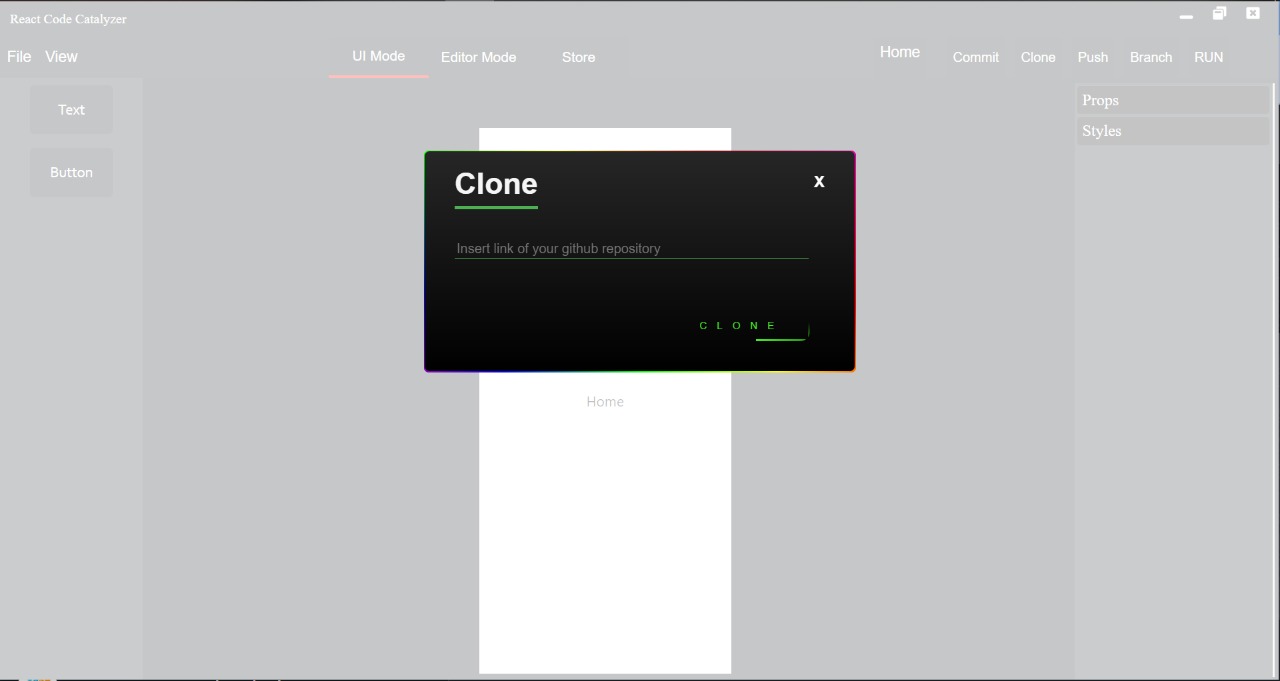


Figure : UI for git clone command

* This commit model allows user to clone a repo from GitHub.

Graphical user interface

Description automatically generated

Figure : UI for git creating new branch command

* This commit model enables user to create a new branch, instead of master(origin).

## Deployment

RCC is working on local host.

# Testing and Evaluation



This chapter includes the manual and automated testing of RCC.

## Unit Testing

**Unit Testing 1:** Managing Projects

**Testing Objective:** To ensure no malfunction while creating applications.

**Test Case Id:** M1-001

**Test Case Description:** Test the Creating App functionality.

**Test Scenario:** Verify on entering valid name.

Table : Test Cases for Managing App

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify that project is being created. | TestProject1 | Successfully created | As Expected | Pass |
| 2. | Verify that every project has distinctive name. | TestProject1 | Notifying user an application with this name already existing | As Expected | Pass |
| 3 | Open an existing application | TestProject1 | Showing in the directory | As Expected | Pass |
| 4 | Renaming an existing application. | TestProject1  Asad | Renamed as “Asad” | As Expected | Pass |
| 5 | Verify that a project is being deleted | TestProject1 | Project successfully deleted. | As Expected | Pass |

**Unit Testing 2:** Managing Modes

**Testing Objective:** To ensure no malfunction while switching modes.

**Test Case Id:** M1-002

**Test Case Description:** Test the Mode functionality.

**Test Scenario:** Verify change of mode.

Table : Test Cases for Managing Modes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify that user can switch onto coding mode. | None | Successful switching | As Expected | Pass |
| 2. | Verify that user can shift to drag n drop mode of the same project. | None | Successful switching | As Expected | Pass |
| 3 | Make changes to the application from either mode. | TestProject1 | Changes reflected on both interfaces. | As Expected | Pass |

**Unit Testing 3:** Check Auto Save

**Testing Objective:** To ensure the work of the user is being saved after every 5 seconds.

**Test Case Id:** M1-003

**Test Case Description:** Test the Real-time saving.

**Test Scenario:** Verify changes are saved.

Table : Test Cases for Auto Save

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify that development is auto saved without pressing ctrl + S. | Home | Successfully saved. | As Expected | Pass |

**Unit Testing 4:** Downloading Components

**Testing Objective:** To ensure that the components can be downloaded from the webstore and then used in RCC.

**Test Case Id:** M2-001

**Test Case Description:** Test the components functionality.

**Test Scenario:** Verify that components can be downloaded.

Table : Test Cases for Downloading Components

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if a component is downloaded. | None | Component is downloading. | As Expected | Pass |
| 2. | Verify that downloaded component can be used in the project. | None | Component visible and functional when dragged on the screen. | As Expected | Pass |
| 3. | Verify Component working on PD | None | Component functional on personal mobile. | As Expected | Pass |

**Unit Testing 5:** Downloading Templates

**Testing Objective:** To ensure that the templates can be downloaded from the webstore and then used in RCC.

**Test Case Id:** M2-002

**Test Case Description:** Test the templates functionality.

**Test Scenario:** Verify that templates can be downloaded.

Table : Test Cases for Downloading Templates

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if a template is downloaded. | None | Template is downloading. | As Expected | Pass |
| 2. | Verify that downloaded template can be used in the project. | None | Template visible and functional when dragged on the screen. | As Expected | Pass |
| 3. | Verify template working on PD | None | Template functional on personal mobile. | As Expected | Pass |

**Unit Testing 6:** Adding Screens

**Testing Objective:** To ensure that multiple screens are being added.

**Test Case Id:** M3-001

**Test Case Description:** Test the screens.

**Test Scenario:** Verify that screens are being added.

Table : Test Cases for Adding More than 1 screen

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if a screen is added | Screen2 | Screen visible on the sw. | As Expected | Pass |

**Unit Testing 7:** Stack Navigation

**Testing Objective:** To ensure that the screens are navigating with stack.

**Test Case Id:** M3-002

**Test Case Description:** Test the stack navigation on screens.

**Test Scenario:** Verify that the screens are moving using stack.

Table : Test Cases for Stack Navigation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if screens are moving using the principle of stack. | None | Screens are stacked on top of one another | As Expected | Pass |

**Unit Testing 8:** Tab Navigation

**Testing Objective:** To ensure that the screens are navigating with tab navigator.

**Test Case Id:** M3-003

**Test Case Description:** Test the tab navigation on screens.

**Test Scenario:** Verify that the screens are moving using the bottom tab.

Table : Test Cases for Tab Navigation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if screens are moving using the bottom tab. | None | Screens are navigating | As Expected | Pass |

**Unit Testing 9:** Git-Hub Cloning Project

**Testing Objective:** To ensure that a project can be cloned from GitHub.

**Test Case Id:** M4-001

**Test Case Description:** Cloning from GitHub.

**Test Scenario:** Verify that the project is added in local machine.

Table : Test Cases for cloning from GitHub

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the project is cloned with the press of a button | None | Project is cloned | As Expected | Pass |

**Unit Testing 10:** Git-Hub pulling changes

**Testing Objective:** To ensure that the changes are pulled from GitHub.

**Test Case Id:** M4-002

**Test Case Description:** Pulling from GitHub.

**Test Scenario:** Verify that the changes are pulled and reflected on the project.

Table : Test Cases for pulling from GitHub

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the changes are pulled | None | Pulled changes reflected on project | As Expected | Pass |

**Unit Testing 11:** Git-Hub pushing changes

**Testing Objective:** To ensure that the changes are pushed to GitHub.

**Test Case Id:** M4-003

**Test Case Description:** Pushing to GitHub.

**Test Scenario:** Verify that the changes are pushed and reflected on the project.

Table : Test Cases for pushing to GitHub

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the changes are pushed | None | Pushed changes reflected on project | As Expected | Pass |

**Unit Testing 12:** Git-Hub add changes

**Testing Objective:** To ensure that the changes are added to GitHub.

**Test Case Id:** M4-004

**Test Case Description:** Adding to GitHub.

**Test Scenario:** Verify that the changes are added.

Table : Test Cases for adding to GitHub

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the changes are added. By running git status command | None | Changes are tracked | As Expected | Pass |

**Unit Testing 13:** Git-Hub commit changes

**Testing Objective:** To ensure that the changes are committed to GitHub.

**Test Case Id:** M4-005

**Test Case Description:** Committing to GitHub.

**Test Scenario:** Verify that the changes are committed to GitHub.

Table : Test Cases for committing to GitHub

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the changes are committed by running git status command. | None | Changes are staged on branch. | As Expected | Pass |

**Unit Testing 14:** Open the Terminal

**Testing Objective:** To ensure that the terminal is functional.

**Test Case Id:** M5-001

**Test Case Description:** Checking the functionality of terminal.

**Test Scenario:** Verify that the terminal is functional and operational.

Table : Test Cases for Terminal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the terminal is functional | None | Terminal is functional. Commands can be written and run. | As Expected | Pass |

**Unit Testing 15:** Running git commands

**Testing Objective:** To ensure that the terminal is functional for git commands.

**Test Case Id:** M5-002

**Test Case Description:** Checking the functionality of git commands on terminal.

**Test Scenario:** Verify the git commands.

Table : Test Cases for Terminal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the terminal is functional for git commands | **git config**  **git init**  **git clone**  **git add**  **git commit**  **git diff**  **git reset**  **git status**  **git rm**  **git log**  **git show**  **git tag**  **git branch**  **git checkout**  **git merge**  **git remote**  **git push**  **git pull**  **git stash** | Terminal is functional for git commands . Commands can be written and run. | As Expected | Pass |

**Unit Testing 16:** Running windows commands

**Testing Objective:** To ensure that the terminal is functional for windows commands.

**Test Case Id:** M5-003

**Test Case Description:** Checking the functionality of windows commands on terminal.

**Test Scenario:** Verify the windows commands.

Table : Test Cases for Terminal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the terminal is functional for windows commands | Assoc.txt  cipher  ipconfig  ping  systeminfo  taskkill  tasklist | Terminal is functional for windows commands . Commands can be written and run. | As Expected | Pass |

**Unit Testing 17:** Running npm commands

**Testing Objective:** To ensure that the terminal is functional for npm commands.

**Test Case Id:** M5-004

**Test Case Description:** Checking the functionality of npm commands on terminal.

**Test Scenario:** Verify the npm commands.

Table : Test Cases for Terminal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if the terminal is functional for npm commands | npm install  npm list -g –depth=0  npm-windows-upgrade  npm outdated -g –depth=0  npm run  npm install -g npm@latest  npm list  nvm install 12nvm-install-latest-npm | Terminal is functional for npm commands . Commands can be written and run. | As Expected | Pass |

**Unit Testing 18:** Drawer Navigation

**Testing Objective:** To ensure that the screens are performing consistent drawer navigation.

**Test Case Id:** M3-004

**Test Case Description:** Test the drawer navigation on screens.

**Test Scenario:** Verify that the screens are moving using drawers.

Table Test for drawer navigation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Test Data** | **Expected Result** | **Actual Result** | **Pass/Fail/Not Executed/ Suspended** |
| 1. | Verify if screens are moving using the drawer principle. | None | Selected screen from the drawer appears to be the front the screen on the application. | As Expected | Pass |

## Functional Testing

The functional testing of each module of RCC is listed below:

**Functional Testing 1:** Login with multiple user ids.

**Testing ID**: 1-M1.

**Objective**: To ensure that the correct user profile gets loaded and relevant projects are displayed and functional.

Table Test script for Module 1-Managing Projects

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Login as a ‘User1’ (already holds an account). | Username: (correct username user1)  Password:  (Correct password 1234) | The correct user page is showcased.  All user projects displayed.  The user can add new projects with unique names.  And the user can shift modes. | Logged in and redirected to the user1 profile. | Pass |
| 2. | Login as a ‘user2’ (do not possess an account). | Username: user2  Password:  1234 | SW prompts an error message suggesting creating an account first. | Login failed – invalid credentials error | Fail |

**Functional Testing 2:** Working of template and components.

**Testing ID**: 2-M2.

**Objective**: To ensure that no component/template is malfunctioning.

Table 121 Test Script for component testing.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Selecting a component i.e., **Touchable Opacity**. | Adding the following props:  Padding: **‘15px 32px’**  Border Radius: **‘10px’**  Border Width: **‘0.5px’**  Color: ‘white’  Back-ground Color: **‘green’**  Align: **‘center’**  Font-size: **‘16px’** | Button created. | Button with these properties and fully functional. | Pass |
| 2. | Selecting a component i.e., **Text**. | Adding the following props:  Color: **‘black’**  Align: **‘center’** | Text with the following properties. For simple paragraphing and description. | Text successfully written. | Pass |
| 3. | Selecting a component i.e., **Text field**. | Adding the following props:  width: **100%**; height: **150px**; padding: **12px 20px**; box-sizing: **border-box**; border: **2px solid #ccc**; border-radius:**4px**; background-color: **#f8f8f8**; resize: **none**; | Text field created | Text field fully functional. | Pass |
| 4. | Selecting a component i.e., **Main Title**. | Adding the following props:  padding: **60px**; text-align **center**; background: **#1abc9c**; color: **white**; font-size: **30px**; | Main heading text styled. | Main heading text editable and styled. | Pass |
| 5. | Selecting a component i.e., **Subtitle**. | Adding the following props:  padding: **40px**; text-align **center**; color: **dark grey**; font-size: **20px**; | Subtitle styled. | Subtitle text editable and styled | Pass |
| 6. | Selecting a component i.e.  **Card** | Adding the following props:  Box-shadow: **0 4px 8px 0 rgba(0,0,0,0.2)**  Border-radius: **5px 5px 0 0**  Padding: **2px 16px**  Width: **40%** | Card with the following properties, suitable for e-commerce and food ordering websites. | Fully functional and editable card. | Pass |
| 7. | Selecting a component i.e.  **Alarming button.** | Adding the following props:  Background color: **‘red’**  Border: **‘none’**  Color: **‘white’**  Padding:**’15px 32px’**  Text-align: **‘center’**  Display: **‘inline-block’**  Font size: ‘**16px**’ | Button made. | Button with these properties and fully functional. | Pass |
| 8. | Selecting a component i.e.  **Neon button.** | Adding the following props:  Height: **35px**  Width: **100px**  Border-radius: **5px**  Position: **‘fixed’**  Top: **48vh**  Display: **‘flex’**  Align-items: **‘center’**  Justify-content: **‘center’**  Left: **35vw**  Box-shadow: **inset 0 0 18px #fff,**  **Inset -6px 0 18px #f3bad6,**  **Inset 6px 0 18px #0ff,**  **Inset -6px 0 30px #f3bad6,**  **Inset 6px 0 30px #0ff,**  **0 0 18px #fff, 4px 0 18px #f3bad6, -4px 0 18px #0ff;** | Sparkly button made. | Button with these properties and fully functional. | Pass |
| 9. | Selecting a component i.e.  **Login with fb button.** | Adding the following props:  box-sizing: **border-box**;  position: **relative**;  margin: **0.2em**;  padding: **0 15px 0 46px;**  border: **none**;  text-align: **left**;  line-height: **34px**;  white-space: **nowrap**;  border-radius: **0.2em;**  font-size: **16px**;  color: **#FFF**;  background-color: **#4C69BA**;  background-image: **linear-gradient(#4C69BA, #3B55A0);**  text-shadow: **0 -1px 0 #354C8C;** | Facebook Login Button, successful and operational. | Fully functional button | Pass |
| 10. | Selecting a component i.e.  **Login with google button.** | Adding the following props:  box-sizing: **border-box**;  position: **relative**;  margin: **0.2em**;  padding: **0 15px 0 46px;**  border: **none**;  text-align: **left**;  line-height: **34px**;  white-space: **nowrap**;  border-radius: **0.2em;**  font-size: **16px**;  color: **#FFF**;  background-color: **#5B7BD5;**  background-image: **linear-gradient(#5B7BD5, #4864B1);**  background: **#DD4B39;**  border-right: **#BB3F30 1px solid;** | Google Login Button, successful and operational. | Fully functional button | Pass |
| 11. | Selecting a template. | None. | Placing the template on the canvas. The text fields and certain cards are editable. Rest of the app is fully functional. | Fully functional with all properties. | Pass |

**Functional Testing 3:** Complete mobile application developing with building screens and performing navigation.

**Testing ID**: 3-M3.

**Objective**: To ensure that the user can build screens and perform navigation with those screens.

Table Test script for Module-3 Developing the app

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Building Screens | Screen Name: **“NewScreen1”**  Operation: **“Add Screen”**  Operation: **“Delete Screen”** | A screen is added to the project.  Naming NewScreen1.  Components and templates can be used on the screen.  The screen can be deleted from the project | Fully functional screen. Components placed on the screen are editable. The user can switch between both modes in that screen. The screen can further be deleted if not needed. | Pass |
| 2. | Performing Navigation on Screens. | None | The user can perform:  Tab Navigation  Stack Navigation  Drawer Navigation  Smoothly with the screens. | All navigations are performed with smooth and consistent running. | Pass |

**Functional Testing 4:** Performing version controlling on the project.

**Testing ID**: 4-M4.

**Objective**: To ensure that the user can perform version controlling on the project.

Table Test script for Module-4 performing version controlling

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Using the buttons to perform all version controlling commands. | None | Pressing buttons on the top bar:  Clone- Window pops that asks the user to enter the project that is needed to be cloned from GitHub.  Pull- Window pops that allows the user to pull from git after user confirmation.  Add- Window pops that prompts the user to confirm adding work.  Commit- Window pops that asks the user to title the changes that are needed to be made in the repo. | Clone- Projects are cloning from git.  Pull- Changes made by other users can be pulled from git.  Add- Changes made can be added in the project.  Commit- Changes made are being successfully committed on GitHub. | Pass |

**Functional Testing 5:** Working with the terminal.

**Testing ID**: 5-M5

**Objective**: To ensure that the terminal is working properly.

Table Test script for Module-5 Integrated Terminal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Running git command. | **git config**  **git init**  **git clone**  **git add**  **git commit**  **git diff**  **git reset**  **git status**  **git rm**  **git log**  **git show**  **git tag**  **git branch**  **git checkout**  **git merge**  **git remote**  **git push**  **git pull**  **git stash** | All commands are running and performing their relevant output | Each and every command was tested separately. Commands providing correct output. | Pass |
| 2. | Running windows command. | ssoc.txt  cipher  ipconfig  ping  systeminfo  taskkill  tasklist | All commands are running and performing their relevant output | Each and every command was tested separately. Commands providing correct output. | Pass |
| 3. | Entering text other then a command. | Entered Text: ‘abd\_009’ | Terminal not operating. | Terminal not showing any output | Fail |
| 4. | Running npm command. | npm install  npm list -g –depth=0  npm-windows-upgrade  npm outdated -g –depth=0  npm run  npm install -g npm@latest  npm list  nvm install 12nvm-install-latest-npm | All commands are running and performing their relevant output. | Each and every command was tested separately. Commands providing correct output. | Pass |

## Business Rules Testing

As there are no business rules devised for RCC. Therefore, business rules testing is N/A.

## Integration Testing

Integration tests for RCC were performed in order to ensure consistency and smooth work flow when user interacts with the sw.

**Integration Testing 1:** Working with the project.

**Testing ID**: IT1-M1.4.5

**Testing Objective:** To ensure the project is being updated to respective changes correctly and *the* ***interface*** *reflects those changes on the canvas*.

Table 125 Test Script for Project Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Create Project | TestProject1 | Successfully created react native project. | Project Created. | Pass |
| 2. | Open Project. | Open a previous project named “TestProject1” | User resumes project from last saved changes. | Project resumed from same place. | Pass |
| 3. | Using Template | TestTemplate1 | “TestTemplate1” integrated to the project “TestProject1” | Successful integration of selected template to selected project | Pass |
| 4. | Release APK | TestProject1 | Release APK generated for “TestProject1” successfully. | Android Package  Kit for release generated. | Pass |
| 5. | Version Controlling | Push  Add  Commit | The project code added to git-repository. Changes successfully committed and the code was pushed. | Smooth and corelated version controlling system performed successfully. | Pass |
| 6 | Commanding through terminal | npm init “TestProject1CommandLine” | Generating a build APK for a react project with command line. | Android Package  Kit for build generated. | Pass |

**Integration Testing 2:** Working on Modes and interaction with Marketplace.

**Testing ID**: IT1-M2

**Testing Objective:** To ensure that the modes are compatible and working parallel. When components are added *the* ***interface*** *between marketplace* and the *two modes is running correctly.*

Table 126 Test Script for Interaction between modes and marketplace

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Adding components. | Text Field  Button | Successfully download button and text field. | Text Field and Button downloaded | Pass |
| 2. | Working on UI Mode. | Adding the “Button”  and “Text Field” to the canvas. | “Button” and “Text Field”. added. Props are alterable- adjustable. | “Button” and “Text Field” visible on canvas. Props can also be changed. | Pass |
| 3. | Working on Coding Mode. | Add a “Text” | “Text” added successfully. | “Text” visible on canvas | Pass |
| 4. | Shifting from UI to Coding. | - | The code for all components and there styling shown. | User can see the entire working in form of a code | Pass |
| 5. | Shifting from Coding to UI. | - | The changes made through coding seen on the canvas in UI. | All changes made are visible on the canvas. | Pass |

**Integration Testing 3:** Connecting with screens

**Testing ID**: IT1-M3

**Testing Objective:** To ensure the screens are linked to one another *and running correctly*.

Table 127 Test Script for Screen Navigation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Creating Screens | “TestScreen1”  “TestScreen2” | Successfully create two screens. | Screens created. | Pass |
| 2. | Navigation of screens | - | Successfully applying navigation on screens to link them and sequence them in the required order. | Navigation successfully done | Pass |

**Integration Testing 4:** Placing components on the screen.

**Testing ID**: IT2-M3

**Testing Objective:** To ensure the components are placed successfully on the canvas *and running correctly*.

Table 128 Test Script for interaction of components with screen

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Actual result** | **Result** |
| 1. | Creating a screen | “TestScreen1” | Successfully created screen. | Screen created. | Pass |
| 2. | Adding a Component from Market place. | - | Successfully adding any component that was downloaded from marketplace. | Button successfully added to screen. | Pass |

# Conclusion and Future Work

This chapter concludes the project and highlights future work.



## Conclusion

RCC has enabled developers to make React Native project with a few clicks. This software aids in building apps to boost your business and provide great opportunities. Users can develop apps with no prior knowledge to any language may it be JavaScript or react native. You just drag it; you drop it, and your application is developed. No need of extensive debugging. Creating applications with just a few clicks, time saving and yet cost effective.

## Future Work

RCC needs some finishing touches. After those the software will be released in the market. We are looking forward to pitching it to some investors (and we have already pitched to some in the past) and making RCC a great success soon.

We have pitched RCC at the SSBC where it has been shortlisted for furthering it as a business idea. Moreover, RCC is also selected among the top 6 projects in NIB city level, in the disciple of e-commerce.

# References

**World Wide Web**

Bespin. “Skywriter And Ace”. Internet: <https://ace.c9.io/> , [May 2, 2020]

Alenzia,“InteractiveAppBuilder”.Internet: <https://docs.interactiveappbuilder.com/docs/introduction.html>, 2020 [Oct 22, 2020]