

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Mobile Mandi: Agricultural Market Intelligence for India

Author:
Jaskirat Singh Chahal

Supervisor:
Anandha Gopalan

Second Marker:
Naranker Dulay

Submitted in partial fulfillment of the requirements for the MSc degree in
Computing Science of Imperial College London

September 2021

Abstract

The provision of accurate and timely market information in low-income countries is currently an underserved latent need of consumers and producers alike. Transaction costs arising from information asymmetries between agents causes market inefficiencies and suboptimal market outcomes. The presence of intermediaries in Indian agricultural markets causes farmers to receive less than 50% of the retail value of their crops. Giving farmers actionable information reduces costs of price discovery and increases their bargaining power.

With this motivation, this report outlines the design, implementation and evaluation of 'Mobile Mandi', a mobile application for agricultural market intelligence. The application presents localised market prices of 22 crops at 157 Indian markets with data sourced from India's National Informatics Centre. Designed for low-literacy, the application provides speech-based interfaces to facilitate information discovery in English, Hindi, Punjabi and its transliterations. Next-day price forecasts were generated by training stacked long short-term memory (LSTM) models on a univariate time series dataset of daily crop prices over ten years. Averaged across a subset of 30 markets and 22 crops, root mean squared error was 1.43 and mean accuracy was 97.9%. Market price forecasts help farmers identify when and where to sell their crops. Usability testing remotely conducted with Punjabi farmers showed promising results. The application was published to the Google Play Store.

Acknowledgments

I would like to express my sincere gratitude to Professor Anandha Gopalan for the opportunity to carry out this research project and without whom this would not have been possible. I would also like to acknowledge Dr Naranker Dulay whose valuable comments informed the design of this project.

Lastly, I would like to thank my family for their support and encouragement.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Aims & Objectives	1
1.3	Contribution	2
2	Background	3
2.1	Access to Information	4
2.2	Mobile Landscape	5
2.3	Literacy	6
2.4	Agricultural Market Reforms	7
2.5	Summary	8
3	Literature Review	9
3.1	Design for Rural Communities	9
3.2	Design for Low-literacy	10
3.2.1	Input	11
3.2.2	Output	13
3.3	Existing applications	13
3.3.1	Reuters Market Light	14
3.3.2	Kisan Suvidha	14
3.3.3	Mandi Bhav	14
3.4	Summary	15
4	Design	16
4.1	Personas	16
4.2	Requirements	19
4.2.1	Problem Statement	19
4.2.2	Vision Statement	19
4.2.3	Design Requirements	19
4.3	Architecture	20
5	Implementation	23
5.1	Database	23
5.1.1	Data Sources	24
5.1.2	Web Scraper	24
5.1.3	Disadvantages of NoSQL	25

5.2	Android	26
5.2.1	Launch	26
5.2.2	Crops	26
5.2.3	Compare	30
5.2.4	Markets	30
5.2.5	Map	31
5.2.6	Chatbot	31
5.2.7	Languages	36
5.2.8	Text-to-Speech	38
5.3	Price Forecasting	38
5.3.1	Long Short-Term Memory Models	38
5.3.2	Model Setup	39
5.3.3	Results	40
6	Evaluation	42
6.1	Heuristics Evaluation	42
6.2	Usability Testing	45
6.2.1	Setup	45
6.2.2	Results	47
6.3	Stress Testing	50
6.4	Android Studio Profiler	50
6.5	Google Play Store	51
7	Conclusion	53
7.1	Limitations	54
7.2	Ethical and Professional Considerations	55
8	Future Work	57
8.1	Data	57
8.1.1	Crops	57
8.1.2	Markets	58
8.1.3	Prices	58
8.2	Languages	58
8.3	Price Forecasts	58
A	System Usability Scale	72
B	Post-Study System Usability Questionnaire	73
C	Usability Evaluation Questionnaire	78
D	Product Reaction Cards	79

Chapter 1

Introduction

1.1 Motivation

Farmers in India face high transaction costs due to barriers of access to actionable information. Information on India's mandis (physical agricultural markets) is "highly fragmented, disorganised, intermediated and incomplete" [1]. Information asymmetry between farmers and intermediaries results in market failure deriving from the informed party's ability to leverage superior information in a transaction. Ineffective bargaining in the presence of intermediaries results in suboptimal market efficiency. Providing access to information so that farmers do not have to physically visit markets to discover prices eliminates transaction costs incurred in price discovery [2]. Actionable information that is both timely and accurate helps farmers make informed decisions, resulting in higher revenues. Farmers predominantly rely on other farmers as sources of information. However, information access remains underutilized and there exists a sizeable opportunity to fill this latent need.

Increasing smartphone penetration, especially in rural areas, provides an ideal conduit upon which to deploy an agricultural information platform. With this motivation, Mobile Mandi was developed as an Android app for Indian agricultural market intelligence. Further motivation is discussed in greater detail in the following chapter.

1.2 Aims & Objectives

Technology that reaches the most marginalised communities has the potential to generate the greatest impact on both individual and aggregate outcomes by transforming the way people discover and share information. This project aims to address deficiencies in information access with the development of a mobile application for agricultural information dissemination. Specific aims of this project include:

- To provide low-literate users with an intuitive interface to access localised agricultural market price information.

- To provide users with various input and output modalities for accessibility.
- To provide users with price forecasts to effectively respond to market forces.

1.3 Contribution

This project develops a proof-of-concept mobile application to reduce information asymmetries and minimise transaction costs. The Android app ‘Mobile Mandi’ provides users with an intuitive interface based on research-backed findings and interaction design best practices. It incorporates several features:

- Provides localised daily market prices of 22 crops at 157 markets across India.
- Provides the ability to listen to an entire market’s prices with one-click.
- Displays contextual prices relative to other markets.
- Allows querying of historical prices.
- Allows price comparison between markets sorted by distance, price ascending, and price descending.
- Provides accurate next-day price forecasts.
- Provides integration with Google Maps for geographic market search, browsing and directions.
- Provides access to an intelligent speech-based virtual assistant.
- Provides access to the virtual assistant within WhatsApp.
- Allows content sharing to external social media platforms.
- Provides an intuitive interface requiring minimal literacy.
- Provides translations in English, Hindi and Punjabi.

The rest of the report is organised as follows. Chapter 2 explores the necessary background around the problem-space in India namely, access to information, mobile technology, literacy, and market reforms. Chapter 3 reviews the related literature, notably previous solutions designed for rural and low-literate communities. This chapter also covers important findings discovered in interface design for accessibility. Chapter 4 then outlines the design process and the mobile app architecture. Chapter 5 describes the implementation of various techniques used to build the application. This chapter also functions as a detailed walkthrough of the app’s features and functionality. It describes the design and implementation of the database, application and price forecasting model. Chapter 6 critically evaluates the application’s usability and performance. Chapter 7 concludes with a summary of this project’s achievements, a discussion of the limitations, and ethical considerations. Finally, Chapter 8 presents opportunities for future work.

Chapter 2

Background

The industrial revolution of the 18th century spurred rapid technological progress allowing humanity to escape the Malthusian trap that previously caused our ancestors to perpetually subsist in poverty [3]. Malthus' theory describes the phenomenon where population growth (growing exponentially) outpaces agricultural production (growing linearly) causing war, famine and economic catastrophe [4]. Technological progress was understood to produce only temporary gains in per capita living standards, which would ultimately be erased by a consequential rise in population. The miracle of persistent technological innovation observed since has produced exponential gains in living standards for those that stood to gain from it; first Britain, then Europe and the rest of the developed world [5].

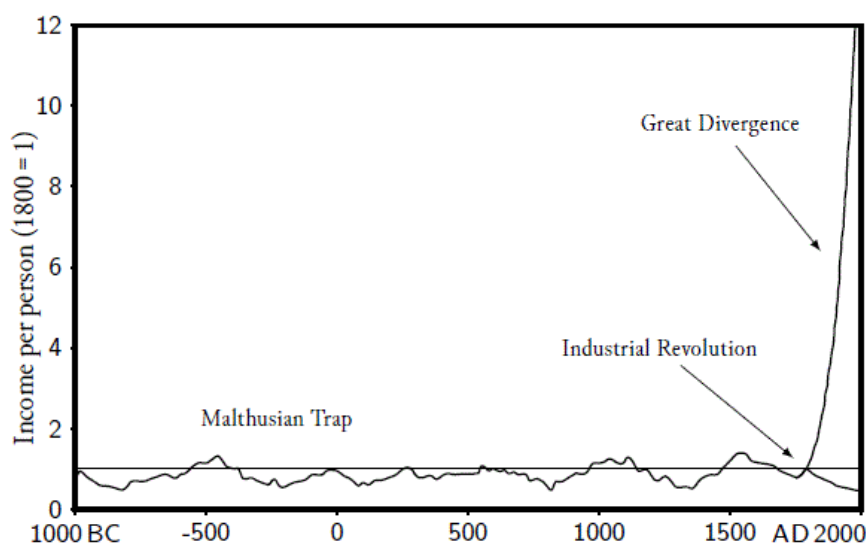


Figure 2.1: Malthusian Escape [5]

Technological progress has similarly revolutionised the transfer of information. An increasingly connected world, powered by the Internet, has virtually eliminated the transmission time of information between netizens located anywhere on Earth. In 16th Century Rome, information travelled at an estimated speed of 1mph (see Figure 2.2) [5, p.306]. In the 21st Century, information travels through fiber optic cables

at the speed of light [6]. Though gains have largely concentrated in the Global North, low-income countries in Africa and Asia are ripe with opportunity for bespoke technological innovations seeking to improve access to information and speed of transfer.

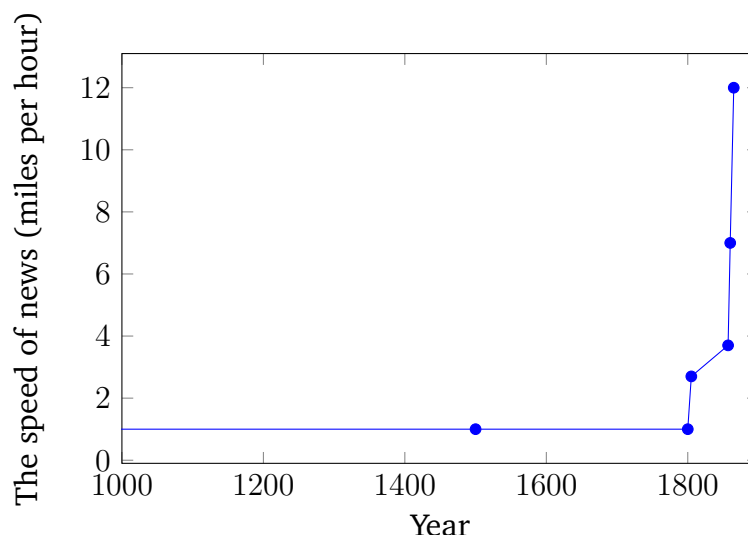


Figure 2.2: Speed at which information travelled [5]

2.1 Access to Information

Improving access to agricultural market information has the potential to improve economic outcomes for the largest subset of India's population. Agriculture is India's largest employer, providing employment to 43% of the workforce [7] and contributing 18% to its GDP [8]. Of these 211 million farmers 86% own less than two hectares [9]. Small agricultural land holders comprise the majority of farmers but in aggregate own just 47% of arable land [9]. In contrast, medium land holders (2 – 10 ha.) comprise just 13% of all farmers but own 44% of arable land [9]. India's 126 million smallholders own an average of 0.6 hectares [9], roughly three-quarters of the size of a football field [10].

One hectare of agricultural land generates, on average, an income of 10 thousand Indian rupees ranging from 6 to 33 thousand [11]. Small land holdings make it difficult for farmers to produce surpluses to financially support themselves and their families. In 2013, 52% of all farming households were indebted with an average outstanding loan of 47 thousand Indian rupees. 53% earned incomes lower than the poverty line [12]. Between 1995 and 2014 nearly 300 thousand farmers committed suicide [13]. Among the reasons cited, bankruptcy, indebtedness, and farming related issues account for nearly 60% of all deaths [14].

India's small agricultural farmers typically depend on intermediaries to sell their crops [15]. Lack of market information creates asymmetries between farmers and

intermediaries in a transaction limiting farmers' ability to bargain effectively [2]. Ineffective bargaining results in lower sale prices and suboptimal revenues. It is estimated that more than 50% of the retail value of farmers' produce is lost to intermediaries [16]. In the economic literature, these are termed search and bargaining costs arising from price discovery and contract negotiation [17]. Access to actionable information that is both timely and accurate can help farmers make informed decisions by eliminating transaction costs associated with search and bargaining [2].

In a 2013 survey of agricultural households in India, just 40% reported seeking access to agricultural information [18]. 18.4% relied on other farmers and 17% reported accessing a media source (see Figure 2.3). This is neither due to irrelevant nor low quality information. Despite the low levels of information access, 96% said information received was useful [18]. Existing information dissemination networks serve as conduits for price discovery. Integration with existing social networks (e.g., WhatsApp) has the potential to magnify reach. 530 million Indians were reported using WhatsApp in 2021 [19], an estimated 38% of the population.

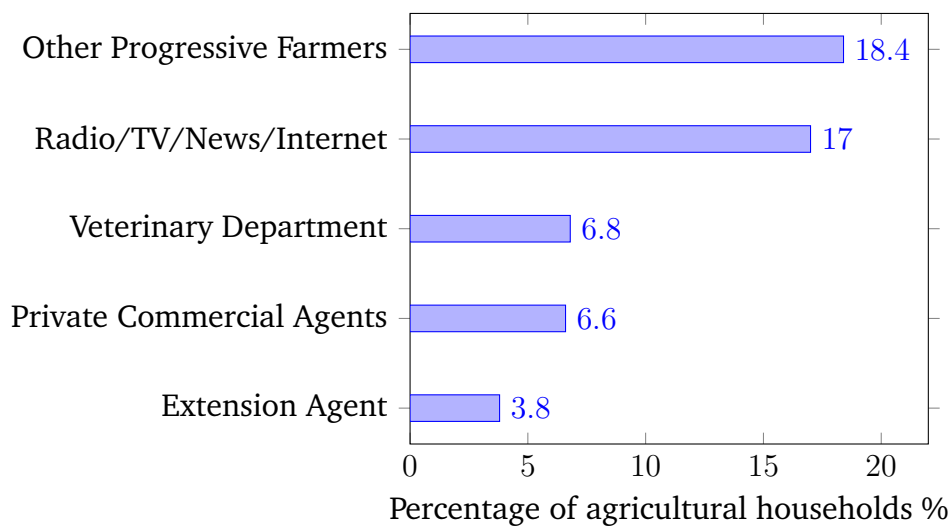


Figure 2.3: Agricultural information sources of farmers [18]

2.2 Mobile Landscape

Since this 2013 survey, the mobile landscape in India has transformed significantly. Fierce competition between Chinese manufacturers Xiaomi, Vivo, Realme, and Oppo has reduced the average cost of a smartphone to around 11,000 Indian rupees [20]. Low-cost smartphones, such as Itel's A48 running Android 10 (Go edition) [21], start at just 5,999 Indian rupees [22]. Alongside falling smartphone prices, wages have also increased. The monthly minimum wage for unskilled agricultural workers in Punjab rose from 5,695 [23] to 9,179 Indian rupees [24] between 2013 and 2020. Cheaper phones and higher wages quintupled smartphone penetration rates from

10% to 55% within the same time frame [25].

Better connectivity and lower network prices have further accelerated this trend. Reliance Jio's dominance of the telecom network has caused a precipitous decline in network connectivity costs. As of early 2020, India's data costs were among the lowest in the world at \$0.2 USD per GB [26]. Unsurprisingly, mobile data usage is amongst the highest in the world with Indian users consuming an estimated 15.7 GB per month [27]. Among its perceived benefits, 92% of mobile users reported smartphones allowed them to "search information that one needs urgently" and "enhances knowledge about the world and others" [28].

Android OS

Android is the dominant mobile operating system worldwide with a market share of 72% with the remainder taken by Apple's iOS [29]. In India, that share rises to a hegemonic 95% [29]. Development for the Android operating system has the potential to reach the greatest number of smartphone users in India and worldwide.

Network Capabilities

Though the 5th-generation wireless network (5G) is set to be implemented within the next few years, users within developing countries often suffer from sporadic network availability and limited network bandwidth. These challenges threaten the user experience and limit the exchange of information on mobile devices. Solutions often involve caching or pre-fetching information while the device is online. Offline-Facebook, for example, demonstrates a Facebook client that pre-fetches content for offline use [30]. To enable seamless user experiences on mobile devices, applications require designs robust to network limitations.

Improvements in smartphone technology, relative to their predecessors, have introduced new interfaces making them accessible to previously marginalised users. Large touchscreen displays have made it easier to perform tasks, such as browsing the internet, taking photos or playing games. Intuitive interfaces and voice assistants provide novel methods of interaction, widening their appeal to low-literate users.

2.3 Literacy

Low-literacy rates unfortunately remain a hallmark of low-income countries [31]. Though contentious debate surrounds the definition, measurement and validity of literacy rates [32, 33], there is considerable consensus on the role of literacy for economic empowerment. Information literate individuals are found to experience greater quality of life derived from their ability to retrieve, evaluate and interpret information effectively [34]. Over the last two centuries, literacy rates have significantly improved worldwide [31]. In 1961, only a quarter of India's population were literate [35]. 57 years later in 2018, almost three-quarters were literate [36]. Paradoxically, the absolute number of illiterate Indians rose from 315 million to 347 million as population growth outpaced increases in literacy. As of the latest census

in 2011, illiteracy rates in rural areas were 32%, double that of urban centres [37].

India's complex stratification of social classes continues to perpetuate inequality of opportunity. Lagging literacy rates among India's scheduled class reveals the disparity (see Figure 2.4) although the difference is narrowing. For context, 20% of all farmers belong to a scheduled caste [9]. These socially marginalised groups experience barriers of access to agricultural information attributable to their caste amounting to an annual opportunity cost of 12 thousand Indian rupees per household [38]. Within India and the developing world, low-literacy remains an important problem and consideration for accessible application design.

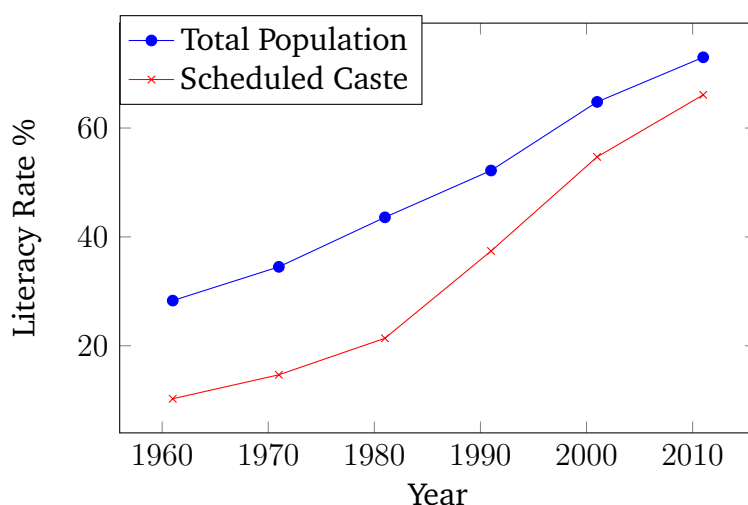


Figure 2.4: Literacy rates in India (1961 – 2011) [35]

Catering to the abundance of languages and dialects is another important consideration for accessible design. Over 270 identifiable languages are spoken in India, grouped by 5 language families [39]. 97% of the population speak one of 22 languages [39]. 44% speak Hindi and 2.7% speak Punjabi [39].

2.4 Agricultural Market Reforms

In 2020, the government of India proposed divisive agricultural policy reforms that threaten farming communities despite promises of improved market efficiency and greater private investment. Uncertainty around the bills' implementation and preservation of minimum support prices (MSP) have led farmers to protest en masse against the reforms. These bills together aim to deregulate agricultural markets nationwide.

1. Farmers' Produce Trade and Commerce (Promotion and Facilitation) Bill, 2020
2. Farmers (Empowerment and Protection) Agreement of Price Assurance and Farm Services Bill, 2020
3. The Essential Commodities (Amendment) Bill, 2020

The first aims to promote within- and between-state trade beyond market premises which would also allow online trading [40]. The second aims to promote contract farming by formalising the contract negotiation process between buyers and sellers [40]. The third aims to promote foreign investment into agriculture [40].

However, in 2006 when similar reforms saw the deregulation of agricultural markets in the Indian state of Bihar, farmers were hamstrung by the lack of accurate information about prices [16]. They incurred higher transactions costs caused by lower bargaining power and received lower output prices relative to other states [16]. Privatisation did not improve infrastructure nor increase private investment [41]. Instead, it led to higher price volatility which negatively affected the stability of farmers' income and lowered crop diversification [41]. In sum, the repeal of the APMC act in Bihar meant that farmers were "left to the mercy of traders who unscrupulously fix a lower price for agricultural produce" [41].

2.5 Summary

Technological progress since the industrial revolution has transformed the speed and volume of information transfer. Increasing smartphone adoption among low-income countries, especially in rural areas, has the potential to improve economic outcomes by providing access to actionable information. In India, a majority of the population is involved directly or indirectly in agriculture. The provision of actionable information can eliminate transaction costs of search and bargaining by minimising asymmetries between farmers and market intermediaries. Empirical evidence reveals that agricultural information sources are currently underutilised by farmers despite their utility. As smartphones have become more affordable, they provide an ideal platform upon which to build a solution for information dissemination. As the dominant operating system, Android's market share provides reach to the greatest number of users. Recent innovations in speech recognition have made it possible for low-literate users, of which farmers contribute disproportionately, to access information previously out-of-reach with entirely speech-based interfaces. Although literacy rates are improving, design for accessibility improves interaction design for all users. Recent deregulation of Indian agricultural markets may promote pernicious price volatility in the long-term. Improving access to market price information may counteract rises in transactions costs and help farmers negotiate effectively.

Chapter 3

Literature Review

There are several areas of related literature that are particularly relevant within this problem space: speech-driven interfaces for developing regions, user interfaces for low-literate users, and technology use in low-income contexts. This chapter presents each topic in turn, evaluating a number of studies and design considerations.

Research in the domain of human-computer interaction for development (HCI4D) reveals successful design requires a deep understanding of the hyper-localised needs and aspirations of the end-user. In the absence of a needs assessment, the next best alternative is to study the findings from previous solutions implemented in the field. With this motivation, I present the result of my findings from several research studies conducted in rural low-literacy developing contexts.

3.1 Design for Rural Communities

The total rural population in 2020 is around 44% worldwide [42]. In India, this number is even larger at around 66%, translating to around 0.9 billion people. A majority of this rural population is either non-literate or semi-literate and often relies on intermediaries to access valuable information. Most low-literate people, disproportionately represented by agricultural workers, interact using voice and rely on TV, radio or public announcements via loudspeaker to deliver locally relevant information, such as health-related warnings. [43].

These traditional sources of information, however, fail to transmit locally-relevant information since they cater to broader locales (eg cities, states) [43]. Solutions such as the VoiKiosk and the SpokenWeb which allowed telephone users to create, deploy and browse VoiceSites using a simple voice interface garnered much success [43, 44, 45, 46, 47].

Challenges & Considerations

These studies noted education and, more broadly, information is inaccessible due to two main factors: 1. lack of affordability of PCs, smartphones and internet access; 2. lack of skill to fully leverage digital technology, due to lack of knowledge [44]. Since the paper was published in 2008, affordability of digital technologies has vastly improved while digital literacy remains poor. These studies highlighted the need for user interfaces to be “radically simplified” while backend infrastructures should be able to support “high volume, low value transactions in the presence of unreliable network connectivity” [44]. Importantly, they mentioned the need for applications to support customisation in terms of content and functionality to cater to user’s local needs [44].

Design for rural communities has the potential to empower a large proportion of the world’s population. Considering projected future trends, mobile technology will increasingly be making it into the hands of previously underprivileged users despite most existing applications designed to cater to an elite subset of the world’s population. To truly empower the next billion users, “they should have the ability to become content creators as opposed to mere content consumers” [44]. Other solutions that have created a decentralised mechanism of information transmission, such as a forum, find that it is the most popular feature within such applications [48].

Avaaj Otalo

Avaaj Otalo, an interactive voice application for small-scale farmers in Gujarat found that their forum for Q&A on agricultural advice was by far their most popular feature [48]. Research suggests that interest in social networking applications comes from an innate desire to belong and the need for self-presentation [49]. When given the opportunity, non-literate users not only understand how to fully leverage the technology but also find innovative uses for it, like turning an advertising board into a platform for social networking [43]. In fact, Avaaj Otalo observed the emergence of social norms, moderation and other innovative features that the system was not intentionally designed to facilitate [48].

The following section will discuss the relevant findings in the field of human-computer interaction for development and user-interface design for low-literacy.

3.2 Design for Low-literacy

One of the greatest challenges in facilitating access to information with technology in developing countries is the lack of information literacy and digital literacy. Over 773 million people in the world are estimated to be completely non-literate [50]. 95% of those live in developing countries and about 70 percent are women [51].

With 313 million illiterate adults, India alone houses over a third of the world's total [52]. People with low levels of literacy were found to primarily use mobile phones for communication only and avoid complex tasks [53]. Users often struggled to use basic functionality such as a contact list for saving phone numbers [54]. Contact list management for illiterate users remains an active area of research. One recent study developed a virtual assistant that helps illiterate users manage their contact list with an entirely speech-based interface [55]. They show that it is possible to develop speech recognition technology for low-resource languages in the Niger Congo family by using the existing abundance of radio broadcasting archives [55].

User Interface Design

Designing mobile user interfaces for low-literacy requires careful consideration of the user's cognitive abilities. Differences in interaction occur due to limited linguistic understanding but more importantly, low-literate users face difficulties with a set of cognitive abilities relevant to digital literacy. These include, "language processing, visual organisation and visual memory, mental spatial orientation, speed of cognitive processing, vigilance, divided attention and perceived self-efficacy" [53]. Design for non-literacy must therefore account for differences in cognitive ability beyond simply the inability to read and write text.

3.2.1 Input

It is found that text-based interfaces are completely unusable for non-literate users and error-prone for semi-literate [53]. Circumventing this problem, developers have designed touch, speech and graphical user interfaces containing little to no text.

Speech

Developers have leveraged speech as a natural form of expression to non-literate and literate alike. Speech circumvents many of the user interface problems that arise when using keyboard buttons or other input affordances that seem unnatural to the novice user. Speech recognition technology has vastly improved over the last decade, however developing automatic speech recognition remains a costly and time intensive endeavor. Low-resource languages such as the local languages typically spoken by non-literates commonly lack annotated speech data for training. Despite these challenges, researchers are able to demonstrate rudimentary speech recognition technology by limiting the size of the vocabulary [56, 55]. Previous solutions that leveraged speech-based input include VideoKheti, Avaaj Otalo and the Spoken Web [57, 48, 45].

VideoKheti is a mobile application designed for agricultural video search in low-literate communities combining speech input with video and audio output [57]. Using a limited vocabulary, researchers were able to design a simple automatic speech recognition feature that mapped user's inputs to interactions within the interface [57]. Speech input provides a familiar and comfortable input modality for

low-literate users, although a major obstacle specific to India is the variety of low-resource local languages and dialects that make it difficult to design general automatic speech recognition (ASR) systems. Such systems typically recognise less than a hundred words and therefore perform simple tasks [55]. A poor performing ASR system has the potential to induce frustration and defeat in users. Therefore poor performance of ASR systems poses a major threat to the success of a voice-based system. As an alternative to speaking single words, users preferred using dedicated numerical buttons to indicate choice from a selection of possible items [43].

Touch

Innovations in touchscreen technology since the mobile phone has seen the disappearance of Nokia, a once domineering presence in the mobile market. Falling smartphone prices have made touch-based user interfaces the norm in entry-level smartphones and are likely familiar to consumers in the developing world. Like speech, touch is another natural form of expression but cognitive differences in low-literate users requires careful UI design. These users show a strong preference towards touch over speech for input, with over 3/4 of successful interactions completed via touch [53]. Results from another study show a preference for multi-page lists over hierarchical menus [58]. Researchers recommend limiting the depth of hierarchical directory structures and keeping navigation linear where possible [53].

Other

In addition to speech and touch input, researchers have also explored other input techniques such as digital pens and styluses that mimic the real world counterpart, providing immediate visual feedback to the user. One such study provided low-literate individuals with “a digital slate prototype that combines natural pen-and-paper interaction with electronic feedback to enhance microfinance record management” [59]. Their solution enhanced the “accuracy of form entries, recording time and data completeness”, however users performed just as well as with an entirely-electronic system [59]. Given the rise in popularity of touch-based digital devices since the paper was published in 2010, it is anticipated that users today have high levels of familiarity and comfort with touch-based digital slates, such as tablets and iPads.

An interesting point to note is that while low-literate users tend to experience difficulty using text-based input, such as keyboards, these users overwhelmingly tend to be numerically literate and prefer using dedicated buttons where possible [60]. Researchers advise against the use of digital buttons whose functions are application dependent as non-literate users find it difficult to remember dynamic functionality [60]. Colours are also helpful in highlighting relevant parts of the user interface. Reds and yellows were found to be especially effective at indicating action [60]. The previously mentioned Avaaj Otalo project relied on numeric touch-tone input to navigate the interactive voice response (IVR) system allowing farmers to ask and

browse agricultural Q&A's [48].

3.2.2 Output

A number of text-free UI's have been proposed for low-literate users. They often use local language text, hand-drawn graphics, photographs, video and animation [53].

Graphics

In KrishiPustak, an audio-visual social networking application for low-literate farmers, all functionality is represented by graphical icons and the interface is entirely text-free [49]. Users have the ability to post photos and videos to a public timeline enabling other users to comment with voice or with photo. Each post is categorised into a specific category: agriculture, livestock and personal, represented by a crop, cow and person icon [49]. The opacity of the selected icon indicates selection.

Farmers accessed the mobile application through one of eight mediators who provided an introduction to the platform and facilitated use [49]. However the role of the mediator is likely to have biased farmers' expectations of the platform, as they often struggled to articulate the use of the system [49]. In contrast to other social networking forums that saw the emergence of norms and moderation, and despite having no knowledge of social networks in the developed world [48], the application promoted more professional networking than social [49].

Researchers behind the KrishiPustak system responded to intermittent network connectivity and limited bandwidth by designing a robust backed infrastructure [49]. Whenever new content was created, it was first stored in a local database while a background process polled for internet connectivity. A background process uploaded content in small incremental units to "maximise short bursts of connectivity" [49]. A manual option to synchronise content was also provided.

Video & Audio

In UIs for low-literate users, audio is commonly used to complement another modality such as graphics or text. Speech has the advantage of lowering intimidation and increasing comfort for digitally illiterate users as it is their natural form of expression. Low-literate users were "thrilled to hear the computer speak in their local language" while using a text-free mobile application [53].

3.3 Existing applications

There are several market information mobile applications currently available for Android. However each has its limitations. Typically they are not designed for low-

literacy and force the user through unnecessary sign-up processes and experience long latency between pages. This section discusses the features and limitations of some existing applications although it is by no means exhaustive.

3.3.1 Reuters Market Light

Reuters Market Light was launched in 2007 to provide agricultural, technical and financial information for farmers in India [61]. It was deployed across 17 states in 9 languages. Featuring a platform for farmers and enterprises to buy and sell agricultural produce, it enabled the circumvention of rent-seeking market intermediaries. The platform grew to 1.3 million registered unique users and is reported to have indirectly impacted another 5 to 6 million farmers [61]. Farmers using the service increased their incomes by 15 to 25% with 73% of those surveyed reporting the application helped them plan their farming activities in advance [61]. This case study provides additional evidence to support the hypothesis that the provision of actionable agricultural information leads to improved economic outcomes.

3.3.2 Kisan Suvidha

Kisan Suvidha is a mobile Android app launched in 2016 by the government of India's Ministry of Agriculture and Farmers Welfare. It provides Indian farmers with market prices, plant protection information, weather and other related features [62]. It is available in 9 languages across India's 29 states and 7 union territories. As of September 2021, it has been downloaded by over 1 million devices. Since it imposes a lengthy sign-up process, however, it is not designed to be accessible to low-literate users. Despite requesting location permissions, the app requires users to manually specify the state, city and specific market at which they would like to view market prices. Users must navigate through deep hierarchical menus and select options within multiple drop-down menus with significant latency between interactions. The app is also unusable offline. These factors all contribute to poor user experience that imposes barriers on low-literate users and discourages repeated use for all users.

3.3.3 Mandi Bhav

Mandi Bhav is a mobile Android app launched in 2020 providing daily news and market spot quotes sourced from the National Commodity & Derivatives Exchange (NCDEX) [63]. Users can receive spot prices on over 500 commodities at over 3000 Indian markets. The app has been downloaded by over 100 thousand devices and it is available in English and Hindi. Although graphics are used for news stories, text is typically overlaid which is unsuitable for low-literate users. Without images of crops, it is impossible to infer which crop prices refer to.

3.4 Summary

A review of the literature unanimously points to the utility of speech-based interfaces for low-literate users. Speech input is a natural method of communication and eliminates the problems associated with text-based interfaces. Poor speech recognition, however, has the potential to induce user frustration. In Avaaj Otalo, researchers experimented with an audible tone to indicate start and end of speech [48]. Speech output thrills users with natural human-like interaction, especially when it speaks their local language. Interfaces must be radically simplified with minimal text and colours must exploit users' prior expectations of meaning. Although low-literate users vary in their fluency, they tend to be numerically literate and interfaces can contain numbers. Information must be localised and customisable in content and functionality. Low-literate users express a preference for multi-page lists with linear navigation over hierarchical menus. Backend infrastructures for rural communities must support high volume, low value transactions in the presence of poor network connectivity. Existing market information applications are not designed to cater to the low-literate user. Poor interaction design, high latency, inadequate use of images and interfaces cluttered with text leave large opportunities for improvement. The next chapter discusses the design and architecture of the application with these considerations.

Chapter 4

Design

Victor Papanek’s seminal work, “Design for the Real World”, describes design as the “conscious effort to impose meaningful order” [64]. In the “Design of Everyday Things”, Don Norman defines it as “the nature of the interaction between people and technology” [65]. Design for mobile applications encompasses both traditional design philosophies and more recent interaction design principles and patterns. Design for low-literacy further magnifies the importance of well-designed interaction.

This chapter begins by presenting a set of personas, created for the purpose of giving clarity to the human-centred rationale behind design decisions. A set of design requirements inspired by the personas are outlined in the following section, in addition to problem and vision statements to guide the design process. Finally the chapter ends with a discussion of the proposed app architecture supported with wire frames (see Figures 4.5-7) and a navigation model (see Figure 4.8) created to guide development.

4.1 Personas

A set of personas were constructed from the research findings contained in the literature review and from my lived experience around agriculture in India. Personas are “composite user archetypes that represent distinct groupings of behaviours, attitudes, aptitudes, goals and motivations observed and identified during the research phase” which are “formalised for the purpose of informing the product design” [66]. They ease the task of understanding users’ goals and help avoid myopic design decisions resulting from a designer’s biases and assumptions. Creating a set of personas helps distinguish how the end-user thinks, behaves, what they want to accomplish and why [66]. Design requirements can then be formulated from these research-backed personas. They ensure design decisions are made with a focus on the end-user’s mental model, as opposed to the implementation model [66]. They also provide a tangible basis for explaining the rationale behind controversial design considerations. By focusing design for the primary persona, secondary and supplemental personas with additional needs are accommodated without encumbering the ability of the product to serve the needs of the primary persona [66].

Name: Mohen

Location: Bihar, India

Primary Language: Hindi

Literacy Level: Low

User Goals: To be better informed

Persona Type: Primary

Background: Mohen is a 55 year old farmer living on the outskirts of Patna, Bihar. He left school at an early age to support his family's income and currently owns a small but sufficient plot of land for farming. He is one of India's many smallholder farmers. He uses his mobile phone primarily to keep in touch with friends and family, using cellular networks and via WhatsApp. Due to his limited literacy, he relies on a small set of memorised operations which were demonstrated when he purchased his smartphone. He occasionally relies on his son for help with unfamiliar tasks on his mobile phone.



Figure 4.1: Mohen [67]

Name: Chamkaur

Location: Punjab, India

Primary Language: Punjabi

Literacy Level: High

User Goals: To get the best price for his crops.

Persona Type: Secondary

Background: Chamkaur is a farmer leader living near Fatehgarh Sahib district, Punjab. He is 40 years old and has been involved in agriculture since leaving school at 18. He comes from generations of farmers who rely on income from staple seasonal crops: sugar cane in winter, wheat in spring, rice in summer and fruits & vegetables in autumn. When each crop is harvested, he takes them to market and hopes market prices are favourable, so that he can continue to invest in new farming techniques, seed varieties and better pesticides. As a farmer leader, he shares advice and information with other farmers over WhatsApp.



Figure 4.2: Chamkaur [68]

Name: Anita

Location: Chandigarh, India

Primary Languages: English, Hindi, Punjabi

Literacy Level: High

User Goals: To maximise profits from arbitrage

Persona Type: Secondary

Background: Anita is a 20 year old affluent market trader living in the metropolitan city-state of Chandigarh in Northern India. She makes a living buying and selling commodities, profiting from price disparities between markets. She often negotiates with market vendors but final prices often depend on a variety of perpetually fluctuating causal factors, including daily variations in supply and demand, crop varieties & quality. She wants to buy crops at a lower price than she sells them and needs a trustworthy source of information to rely on for price information.



Figure 4.3: Anita [69]

Name: Rahul

Location: Madhya Pradesh, India

Primary Language: Hindi

Literacy Level: Medium

User Goals: To keep costs low

Persona Type: Secondary

Background: Rahul is a 30 year old trucker who spends much of his time travelling long distances across India. He is one of the 8.5 million truck drivers supporting the logistics industry, transporting various goods across the country. He travels upto 500 kilometres each day and he rarely returns home to his hometown of Indore, Madhya Pradesh seeing his young family just a few days a month. His truck has become a second-home and is a source of pride. He adorns the interior with decorations and paints the exterior with positive messages like “Horn Please!” and “Go Slow Be Happy”. Rahul hopes to save up for a new art piece and to send money back home for school supplies.



Figure 4.4: Rahul [70]

4.2 Requirements

4.2.1 Problem Statement

Indian farmers and consumers lack access to real-time market information, resulting in sub-optimal market equilibrium with either excess demand or excess supply. Existing mobile platforms are poorly designed for low-literate users with deep hierarchical structures, high latency and limited offline functionality. Improving access to accurate, timely market information will empower users with the knowledge to bargain for fair prices and improve the efficient functioning of agricultural markets in India. Markets that are more efficient are more equitable and help both farmers and consumers meet their profit-maximising goals, by reducing information asymmetries and transaction costs.

4.2.2 Vision Statement

The design of Mobile Mandi will help users achieve optimal profits by improving the price discovery mechanism with greater accuracy, efficiency, accessibility and without the problems of existing platforms that they currently experience. This has the potential to dramatically improve farmers' livelihoods and profitability [71].

4.2.3 Design Requirements

From the preceding personas and statements, a set of design requirements were identified.

- The user must be able to directly view their local market's prices, without any setup or authentication procedure, every time the app is launched.
- The user must be able to visually determine the context of a crop's price.
- The user must be able to compare prices between markets.
- The user must be able to view market information without the need for constant internet connectivity.
- Low-literate users must be able to discover price information naturally using voice-input and receive responses as speech-output.
- The user must be able to change language according to preference.
- The app must conserve precious mobile resources such as battery and cellular data.
- The user must be able to share information with his friends and family.

Research forms the bedrock upon which the persona hypothesis rests and in turn defines ideals of well designed user interaction. The design requirements as outlined form the research-based justifications for design decisions made in the next phase of development: product design & prototyping. These requirements guided the design process in conjunction with the interaction design principles, patterns and best practices as outlined in Alan Cooper's About Face [66]. A hybrid of methodologies were used including Cooper's Goal-Directed Design [66] and Human-Centred Design [72]. Visual design was guided by the Android Material Design guidelines [73].

Ideal user interactions informed by the previous section were translated into a set of design requirements. These requirements set the foundation for brainstorming the visual design and interaction flows through the application. They also informed the technologies required to support functionality. A series of sketches were iteratively developed and interactions story-boarded before formalising the final interaction framework. To minimise cognitive load on users, careful attention was paid to existing interaction flows on popular apps such as WhatsApp, Google Maps and YouTube. These apps served as exemplary frameworks of good interaction design. As any parent will know children, with limited literacy skills and non-existing mental models of technology implementation, effortlessly browse YouTube and find their favourite videos with enviable ease. An interaction design framework mimicking mainstream flows avoids excess effort exerted by low-literate users in memorising seemingly arbitrary sequences of navigation. A user interface that is pleasantly familiar further maximises satisfaction. These benefits are not only exclusive to low-literate users but extend to users of all cognitive and literate ability.

4.3 Architecture

A single-activity architecture was chosen, with the activity acting as the entry point into the application and as a container holding various fragments as they transition in and out of view during navigation. This approach has several advantages over multiple-activity applications:

- Faster UI transitions [74]
- Ability to share scoped data, state and views that are common between screens [75]
- Simplified lifecycle management [76]

Visually the Main Activity contains the app's top action bar [77] and bottom navigation view [78] (see Figure 4.5). A fragment container view [79] is inserted between the two views.

Figure 4.8 presents a simplified navigation flow through the application.

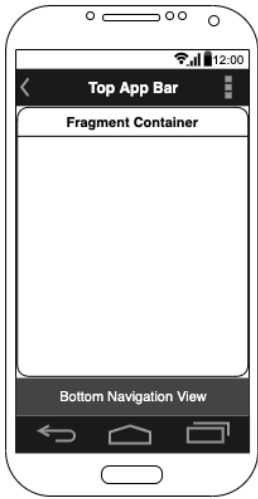


Figure 4.5: Main Activity Wireframe



Figure 4.6: Fragment Wireframes



(a) Closest (b) Low to High (c) High to Low

Figure 4.7: Compare Fragment Wireframes

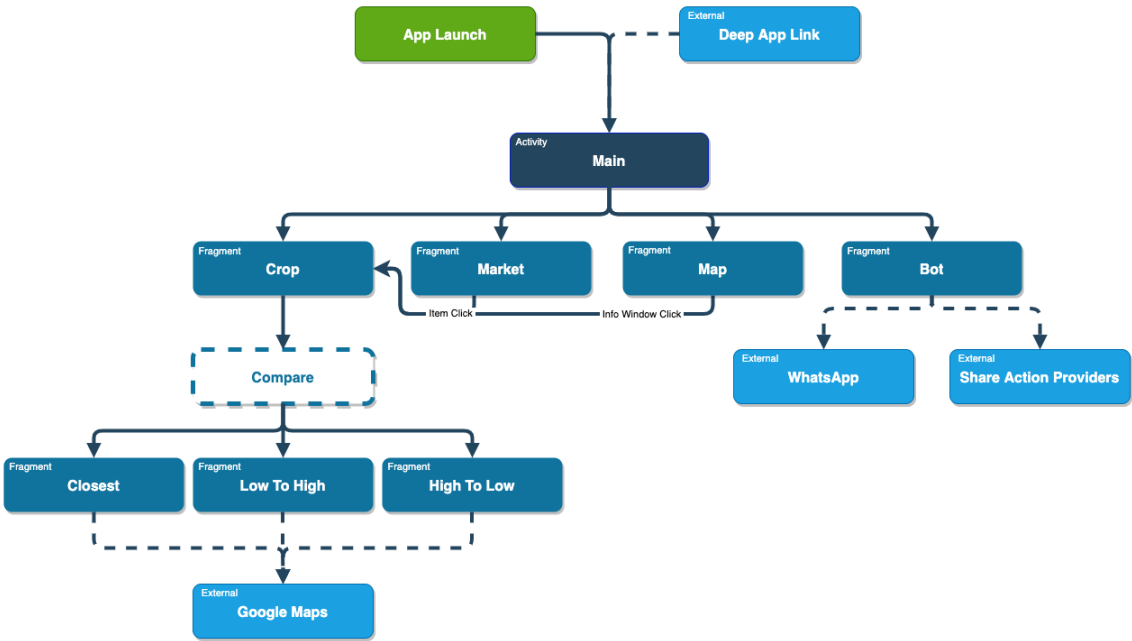


Figure 4.8: Navigation Model

Chapter 5

Implementation

This chapter begins with a discussion of database design, its data sources, and the challenges faced working with NoSQL. The following section gives a walkthrough of the features and functionality comprising the app, from launch to exit. The interaction flow is consistent with the navigation model in the previous chapter. Also showcased is the app's interface design, in particular the features designed for low-literate users. These include the voice-based virtual assistant, speech output, use of semiotics, and language. This chapter then discusses the design, implementation and evaluation of a price forecasting model for next-day price predictions. Results show good predictive accuracy although performance is comparable with the baseline.

5.1 Database

Google's Firebase Realtime Database [80] was selected due to its convenient integration with Android. As a NoSQL database, it offers several advantages and performance benefits over a relational database in the design of Android applications.

- Data persists locally even when the device loses cellular network connection, enabling offline functionality
- Database updates are automatically queued and synchronized when the device returns online

In addition to these benefits, the use of a `FirebaseRecyclerAdapter` further simplifies the process of binding data from the database to the app's UI and listening for live updates [81]. In the absence of Firebase UI, one would need to override the `onChildAdded`, `onChildChanged`, `onChildRemoved`, `onChildMoved`, and `onCancelled` methods of a `ChildEventListener` to listen to live updates. With a `FirebaseRecyclerAdapter`, only two methods: `onDataChange` and `onError` need to be overridden.

5.1.1 Data Sources

Prices

Daily crop prices were sourced from the Price Monitoring Division under the Department of Consumer Affairs hosted by India's National Informatics Centre [82]. Prices of 22 essential crops at 157 markets are reported daily at roughly 5PM Indian Standard Time and can be accessed by inputting fields in an ASP.NET web form. The response is formatted as a HTML table, with market rows, crop columns and price entries in Indian Rupees per kilogram (INR/kg). Data from particular markets is sporadically unreported, in which case the entry is designated by NR for "Not Reported". Country-wide statistics, such as average, maximum, minimum and modal prices are additionally reported by crop.

Coordinates

Coordinates for each market were retrieved using Google's Geocoding API [83]. A list of locations were iterated through to find each market's latitude and longitude. Each market was plotted onto a map and manually checked to verify correctness. These coordinates were stored in the database under `locations`.

States

Alongside coordinate data, each market corresponds to an Indian state. These were similarly found using Google's Geocoding API by parsing the raw response for the top-level administrative area [83]. Locations in disputed territories [84], such as Jammu, Poonch and Srinagar in the state of Jammu & Kashmir, did not offer this information and were manually added.

5.1.2 Web Scraper

Data collection was automated using Selenium WebDriver. Selenium enables scripts written in Python (and other languages) to control the actions performed by a web browser [85]. The script automates the process of acquiring, parsing, structuring and storing the data into the Firebase Realtime Database.

The branch of the database titled `latest` contains the prices of crops that have been stored in the database using the `.update()` method [86]. This method overwrites the values at child nodes only if they are not reported as NR. Since unreported prices do not provide any value for the user, it was deemed better to present last known prices than to present the user with no information.

Each day's data is additionally stored under a node labelled by its timestamp in the form `yyyy-MM-dd`. The child nodes here intentionally include values reported as NR which enables the user to view actual reported prices at historical dates. To avoid type casting errors, entries were stored as `String` types, padded with leading zeros using the `.zfill()` method for correct alphanumeric sorting of `String` types on-device. This avoids errors such as 100 appearing before 99 in a ascending sorted list

```

Crops: {
  Atta: {
    Adilabad: 45
    Agartala: 32
    ...
  }
  Gram Dal: {
    Adilabad: 64
    Agartala: 80
    ...
  }
  ...
}

Markets: {
  Adilabad: {
    Atta: 45
    Gram Dal: 64
    ...
  }
  Agartala: {
    Atta: 32
    Gram Dal: 80
    ...
  }
  ...
}

```

Figure 5.1: NoSQL Database Redundancy

of prices, and vice versa.

Scheduled execution of the web scraping script occurs each day at 23:30 British Summer Time, running on an Imperial standalone virtual machine (VM). The VM runs Ubuntu, a flavour of Linux, that contains the software utility cron enabling the scheduling of repetitive tasks, known as cronjobs, at periodic intervals. Since the VM remains always online, the process of scraping data is entirely automated. To be certain that the script executed successfully, the standard output is emailed to a specified email after execution terminates.

5.1.3 Disadvantages of NoSQL

In hindsight, there were a few major disadvantages of NoSQL relative to a SQL database:

- Difficulty performing complex queries, such as sorting by two conditions
- Necessary redundancy for two-way relationships in the data [87].

To enable fast and lightweight client querying, data was unavoidably duplicated within the database's JSON tree structure (see Figure 5.1). Retrieving a market's prices, for example, required the data to be structured by market at the first level, and then by crop at the second. On the other hand, price comparisons between markets required data to be sorted by crop first and market second. Although redundancy typically presents potential problems for scaling, each month's data occupies a negligible 5MB of storage.

5.2 Android

With over 95% market share, the popularity of Google's operating system in India prompted development of an Android mobile application [29]. Android Studio was chosen as the primary development environment and the integration of Google's Firebase services [88] with Android made it a convenient choice for authentication, database, data storage and cloud functions.

The minimum SDK is API 19 Android 4.4 (KitKat) meaning the app will run on an estimated 98.1% devices. In addition, components were selected from the `androidX` library which provides backwards-compatibility with earlier devices.

5.2.1 Launch

When the app launches, the user is presented with a permissions dialogue asking whether they would like to allow Mobile Mandi to access the device's location. If the user clicks Deny, the latest modal prices in India are displayed. If the user clicks Allow, prices from the closest market are displayed. This is accomplished in the following sequence.

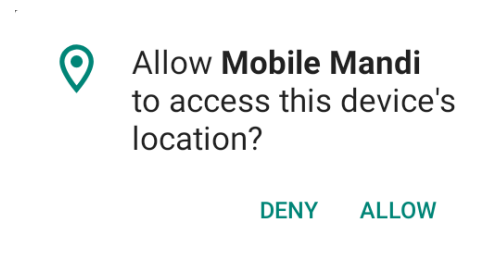


Figure 5.2: Permissions Dialog

1. The user is anonymously authenticated with Firebase, which generates a unique user ID that persists for the duration the app is installed on the device.
2. The database is queried for all the markets and their coordinates.
3. The geographic distances between the location of the device and the coordinates of each market are calculated. During calculation, the market with the minimum distance over previously calculated distances is recorded.
4. A map of markets and distances are stored in the database under `user/{unique_user_id}/`
5. The closest market is passed as a parameter to a new instance of a `CropFragment` which queries the database and populates the view.

5.2.2 Crops

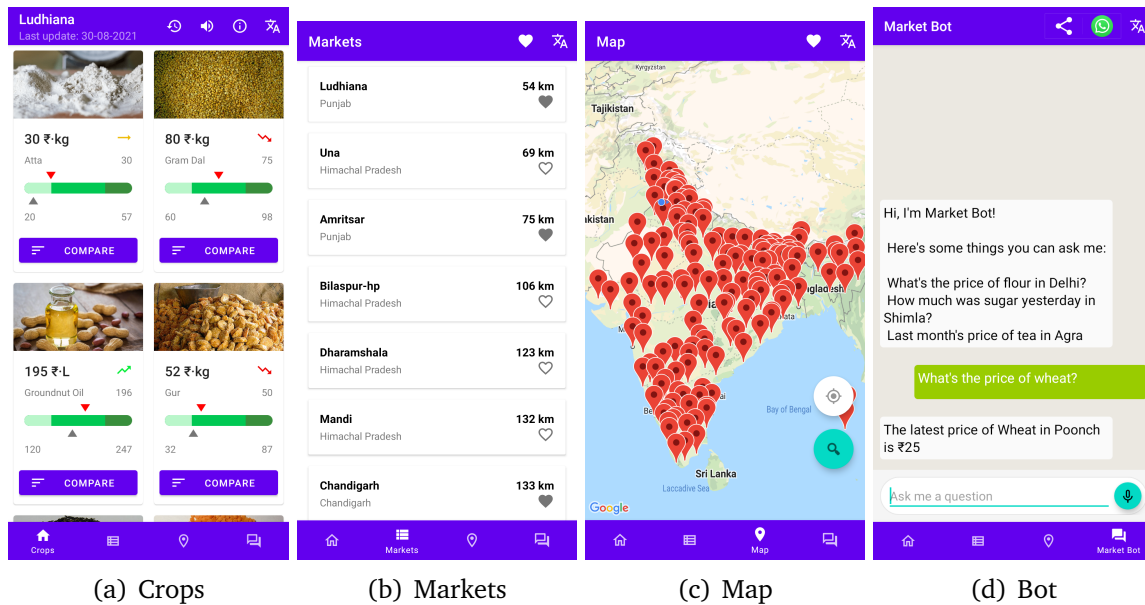


Figure 5.3: Fragments

After the launch procedure completes, the user is taken to the CropFragment (see Figure 5.3(a)).

Images

This screen was intentionally designed to emulate the experience of walking through an Indian market. Creative Commons licensed culturally relevant images were carefully chosen to appear familiar and consequently maximise contextual understanding for Indian users. Large high-resolution JPEG images were initially used which resulted in poor performance from slow rendering. These were replaced by smaller PNG images ideal for mobile use and optimised using ImageOptim [90]. Lossy compression of PNG images at 90% quality was found to reduce file size by roughly 50% on average and from 12MB to 6MB in total. This reduced total application size by almost 25%. Design for the developing world requires efficient use of costly resources such as the cellular data required to download applications. Image optimisation met this need and further resulted in large performance benefits from fast rendering critical for seamless user experience on older and lower-powered devices.



Figure 5.4: Grains at a Delhi market [89]

Images are locally stored in a HashMap on-device, which are included in the application bundle when the app is initially downloaded from the Google Play Store. When requested, the device fetches the correct image resource identifier with the associated crop with time complexity $O(1)$. The HashMap was verified to produce unique

hash codes by checking equality between the size of a HashSet of hash codes with Java's `.hashCode()` method and the number of crops. Zero collisions confirms that retrieval in this case has time complexity $\Theta(1)$.

Prices

Prices alongside their respective units, in Indian rupees per kilogram or litre, are displayed in bold font for easy readability while the user is scrolling and skimming. Crop names are displayed below in a lighter, smaller font to minimise cluttering the display with text. In an early prototype, prices and crops were displayed in the reverse order making reading and skimming difficult, especially on smaller screen sizes. Crops are always positioned in alphabetical order which eases cognitive load as users familiarise themselves with the application and learn spatial mappings.

Linear Gauge Chart

Prices in isolation offer little value. A user's goal is to optimise price received, relative to the surrounding markets. For a farmer, the identified user goal is to maximise sale price while for a consumer, the goal is to minimise purchase cost. Market traders embrace both of these goals: buy low and sell high. It is therefore necessary to provide the context behind a market's prices: whether a crop is cheap or expensive relative to other markets. Having a single-axis, the linear gauge chart provides at-a-glance context for users of all cognitive ability and graph literacy. The user can see minimum and maximum price at each end of the chart alongside the current and modal price indicated in red and grey respectively. The chart itself is gradated with shades of green to indicate low, moderate and high. Linear gauge charts are typically coloured in red, yellow and green but since users can have opposing notions of prices deemed optimal, shades of a more neutral green were preferred.

Difficulty in using existing Android charting libraries, such as AnyChart [91], inspired the development of a custom drawable resource [92]. It was constructed by overlaying three rectangle items within a `<layer-list>` element used to theme and style a linear progress bar. In effect, this creates a gradated linear gauge chart with rounded corners.

At run-time the current and modal price indicators are positioned. This occurs after the view is created, since they are dependent on the width of the progress bar. The width is not known beforehand and cannot be hard-coded since it varies from device to device. This method can occasionally result in a minor flicker when the view is initially populated with new data, but does not appear afterwards since the data is

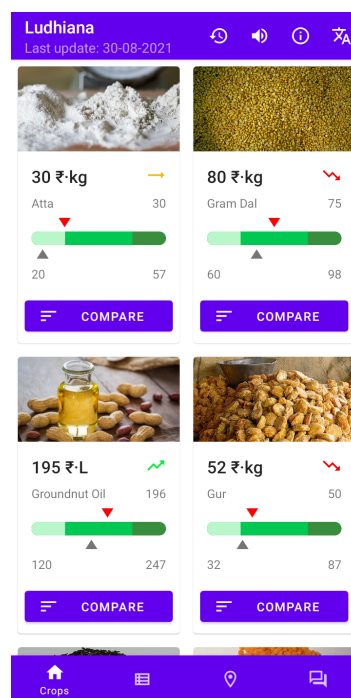


Figure 5.5: Crops

cached.

Help Dialogue

Early feedback found that the linear chart was not immediately intuitive to all participants. Once participants were informed however, they thought it was easy to understand. In response, a help dialogue was added to the top action bar (see Figure 5.6). This unobtrusive dialogue helps the user understand the chart's meaning without cluttering the display once they become comfortable with the interface. A question mark symbol was initially used before resorting to the more universally recognised info symbol, as supported by Microsoft's Windows UX guidelines [93], Google's Android Material guidelines [94] and Apple's human interface guidelines [95].

Date Picker

Users can click on the history icon in the top app bar to view historical prices. On click, a dialogue appears in focus presenting the user with the current date highlighted in the current month. Constraints were added to the calendar to prevent the user from selecting a date in the future, or before 1st June 2021 which is the earliest date in the database. On selection, the database is queried for the specified date and current market and the view is repopulated with the requested data. To return, users can click on the home button in the bottom navigation or click the system back button. This feature exhibits the same behaviour even if the user selects a market other than their home market. Users have complete flexibility to query historical prices at any date and market.

Sound

The sound icon was added to allow users to listen to all of a markets' prices. During observation, users would methodically click each crop's card, activating text to speech, to listen to its price. The addition of this feature streamlined this interaction flow to one click, providing users with an easy way to passively listen. With 22 crops displayed, text to speech takes a long time to finish speaking. "Play" and "Stop" buttons give the user additional control over speech. The icons used for these buttons were selected to mimic the image of a person speaking. The conventional triangle and square buttons were replaced in favour of the former to maximise recognition for low-literate users.

Languages

A translations icon always appears as the right-most icon providing a consistently familiar touchpoint throughout the application. On click, a list of languages are dis-

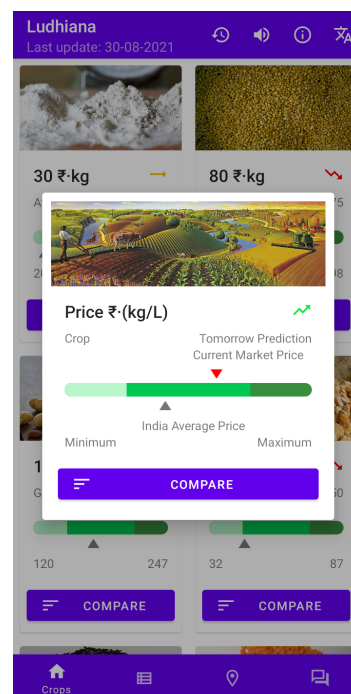


Figure 5.6: Help Dialogue

played in their native script alongside a country flag to help non-literate users discern the appropriate selection. The design of this menu was informed by a discussion on culturally relevant iconography for translations and the provision of multilingual content [96].

5.2.3 Compare

Each crop card contains a COMPARE button which allows the user to compare a single crop's prices between markets. Markets are sorted in descending order by distance, and by ascending and descending order by price. Since users will commonly want to compare prices between the closest surrounding markets, a button click sends the user to the Closest screen. Here, the user sees a list of markets in descending order by distance, in addition to prices. Since the context scenario identified the farmer's goal of finding the market with the best prices to sell, a directions icon is additionally displayed. When pressed, Google Maps opens with directions to the particular market. The user can click on the tab headings to move between fragments or simply use a more intuitive horizontal swipe gesture.

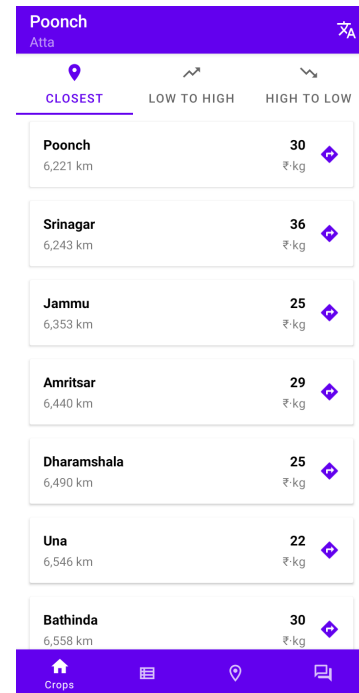


Figure 5.7: Compare

Similar to the way that the Main Activity holds a container for fragments to transition in and out of view, the Compare Fragment holds a ViewPager2 [97] serving as a container for the Closest, LowToHigh and HighToLow child fragments. The Compare fragment is responsible for instantiating the TabLayout and ViewPager2 while the child fragments query the database and populate the RecyclerView with data.

5.2.4 Markets

By clicking the Markets tab in the Bottom Navigation View, the user sees a list of markets in descending order by distance. By clicking on the card, the user is shown a list of prices from the specified market, as before.

Favourites

A click on the heart icon enables the user to save their favourite markets and store them in the icon in the top app bar. Favourite markets are saved in the database under the user's unique user id. When the user quits and opens the app, their favourite markets persist. The icon in the top app bar dynamically responds to users adding

markets to their favourites. When the favourites list is empty, the icon is outlined and when it contains at least one market, it is filled. This provides a subtle hint telling the user the icon is clickable.

Another separate approach to present favourite markets together at the top of the list was only partially successful. Since data in a RecyclerView is not all loaded at once, favourite markets at the bottom of the list did not appear until the user scrolled down to prompt loading. In practice, markets either did not all appear or were not grouped together. This method was found to be unstable and was dropped in favour of the former.

5.2.5 Map

For more experienced users, the map serves as an additional platform allowing the user to search and find markets geographically. For users travelling between cities or for the curious, the map presents an extra method of interaction. Each one of the 157 markets has their own marker which, when clicked, enables the user to view its market prices. Favourite markets from the previous fragment are located in the top app bar which, when clicked, takes the user to that particular market on the map. The white Floating Action Button (FAB) animates the view to the user's location while the teal FAB shows a view of the entirety of India.

5.2.6 Chatbot

Speech-based interfaces were identified in the literature as a crucial medium for low-literate users. They provide a natural input and output modality that likens technology to an anthropomorphic assistant with the potential to lower intimidation and increase comfort for less experienced users. The design of the chatbot was motivated with the aim of promoting the distribution of useful information by lowering the cognitive demands on the user.

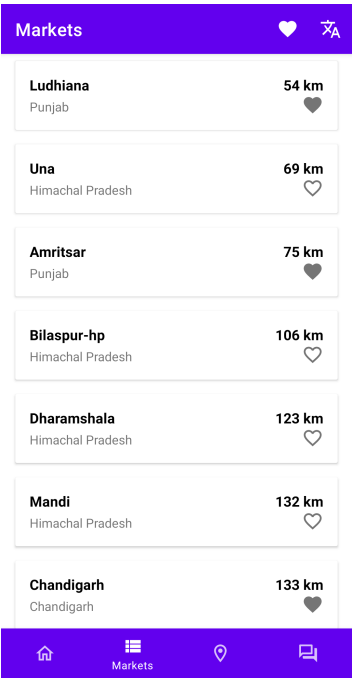


Figure 5.8: Markets



Figure 5.9: Map

The chatbot was built using Google’s DialogFlow platform that enables the embedding of conversational interfaces in a wide range of applications and use cases [98]. When a user sends a request, the Dialogflow agent parses the query text and extracts useful information from pre-defined entities such as the crop, location and time. For example, the request “What was the price of tea in Agra last month” contains the entities “tea”, “Agra” and “last month”. When a question contains at least one of these entities, the agent queries the database and returns a response. If the user does not specify a market, the agent queries prices from the closest market to the user using the unique user identifier generated during authentication. If the user does not specify a date, the latest prices are reported. If the user does not specify a crop, all market prices are returned. In this way, there are 7 unique responses that may be returned to the user, depending on the content of the request. The user may also ask for the closest market, maximum, minimum, modal, or average prices and the date the database was last updated.

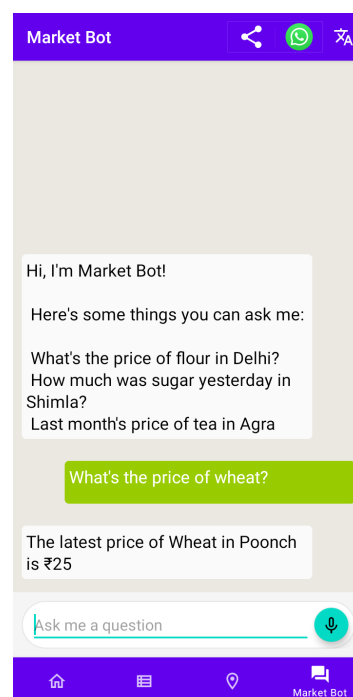


Figure 5.10: Chatbot

The chatbot was intentionally designed to minimise the likelihood of unsuccessful interactions that may lead to user frustration and dissatisfaction. The low-barrier equally streamlines the conversational flow for more experienced users who want immediate access to data they need. Hand-holding an intermediate or advanced user through a long-winded conversational interaction would likely also lead to frustration and dissatisfaction. The current design allows the user to incrementally build their query to more advanced queries without making the agent unresponsive.

In addition, Google Cloud stores a log of all user queries which can be analysed to enable continued enhancements to the fulfillment of user requests. Since the Dialogflow agent exists independently of the mobile app, changes to the fulfillment logic can occur without requiring an app update. Fulfillment of user requests is done via Google Cloud functions in Node.js [99]. Dialogflow’s Inline Editor [100] contains the request fulfillment logic in addition to the mapping between intents and function handlers.

After observing test participants interact with the agent, some were unsure of the types of requests that could be made. To lower potential intimidation, a friendly welcome message was added with examples of the types of questions that can be asked.

Social Media Integration

Existing information dissemination networks provide a crucial mechanism for the propagation of market prices. Given the indisputable popularity of WhatsApp in

India, it was important to provide a seamless way to share information on social media. Users can touch-and-hold on any received message to share to WhatsApp. It is designed as such to leverage the user's expectations of functionality common across messaging apps, including WhatsApp. The user may also tap on the share icon in the toolbar to select various sharing options including email, messages and Bluetooth. Frequently used platforms appear alongside the share icon (see Figure 5.10) further reducing the number of touch actions required to share a message.

WhatsApp Integration

Many Indians commonly associate the Internet with popular social media platforms like YouTube, Facebook and WhatsApp. To reduce the barriers in accessing market information, the Dialogflow chatbot was integrated with WhatsApp so that users can query and share data without installing a new app or leaving WhatsApp. The integration was deployed onto the 3rd-party chat service platform, Twilio, using Google's Cloud Run. Google provides instructions to deploy a Docker container onto its Cloud Platform that runs an Express.js web service connecting the Twilio agent to Dialogflow [101]. WhatsApp integration expands the service to non-Android users. The chatbot can be accessed by Apple iPhone users, for example (see Figure 5.11), who comprise the remaining 3% of the Indian mobile market [29].

The deployment was further modified to recognise and respond to requests in both English and Hindi. Therefore, users can switch language within a single conversation and the agent will automatically respond in the appropriate language. Language recognition is performed by simply checking whether the request text contains characters with Unicode values within the Devanagari Unicode block [102].

Another solution was trialled using Google's Compact Language Detector v3 (CLD3) neural network model for language identification [103]. By spawning a Python3 child process in the web service's `server.js` file, I passed the query text as an argument for the model to asynchronously respond with its language prediction. Although the model was found to have good performance for straightforward inputs, performance degraded significantly in the presence of minor spelling mistakes rendering it unusable for the use case of mistake-prone users typing on their mobile phones.



Figure 5.11: WhatsApp Integration

Android App Links

Each message shared to social media platforms is appended with “Open in Mobile Mandi. `mobilemandi.page.link/app`”. When another user clicks this link, they are redirected to Mobile Mandi’s store listing on the Google Play Store [104]. If the app is already installed, clicking on the link opens the app.

The link is similarly contained in responses from the Twilio-WhatsApp chatbot. The last path segment indicates the market at which prices are related. For example, the link `mobilemandi.page.link/amritsar` with path `amritsar` appears when the user asks for prices in Amritsar. Opening this link prompts the user to install the app from the Google Play Store and once installed opens the respective market. If already installed, the app immediately opens to display market prices in Amritsar.

Firebase Dynamic Links [105] and Android App Links [106] were combined to support this functionality. The subdomain `mobilemandi.page.link/{market}` redirects to `jaskiratchahal.github.io/mobile-mandi/{market}` which, if not installed, redirects to the listing on the Google Play Store. The app recognises the URL and uses an intent filter to send the user to the relevant crop prices page. Firebase Dynamic Links enable the conditional redirect to the Play Store while Android App Links enable intent handling to present the user with the correct data.

The cumbersome manual creation of over 150 Firebase Dynamic Links (one for each market) was circumvented by automating the process with Selenium IDE [107].

Speech input

In addition to text input, speech input allows the user to communicate by speaking into the device’s microphone. Speech-to-Text is performed by Android’s `SpeechRecognizer` service by streaming audio to remote servers [108]. This service is powered by Google Cloud’s Speech-to-Text API and supports several Indian languages including Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Punjabi, Tamil, Telugu and Urdu (as of September 2021) [109]. Hindi and Punjabi are demonstrated in-app.

When the device begins recording, an audible tone is played to indicate it is ready for speech. Text in the message field changes from “Ask me a question” to “Listening...”, prompting the user to begin speaking. Recognition reaches a timeout at the end of speech and after a minimum of 5 seconds, giving the user a generous time window within which to speak. Partial results from the speech recognition appear in the message field while the user is speaking and the final result is automatically sent at the end of speech. Unique tones provide audible user feedback to indicate either successful or unsuccessful recognition.

Speech output

While the welcome message uses on-device Text-to-Speech, subsequent messages

take advantage of Google Cloud's Text-to-Speech API to generate spoken responses. Using DeepMind's WaveNet generative model for synthesis of speech [110], results sound closer to natural human speech and support a wider range of languages than are typically supported by the on-device Text-to-Speech model. Of the Indian languages, Bengali, Gujarati, Hindi, Kannada, Malayalam, Tamil and Telugu are currently supported (as of August 2021) [111]. Audio responses are cached locally allowing the user to tap on a message to hear it spoken aloud again.

Hindi

Dialogflow can support agents in multiple Indian languages including Bengali, Hindi, Marathi, Tamil and Telugu [112]. With Google Translate API, crop and market entities were translated into Hindi and manually verified for correctness. When the user sends a message, the agent detects the set language code and returns the response in the same language. Since Google Cloud Speech-to-Text and Text-to-Speech support Hindi, the user may naturally converse with the agent.

Punjabi

Although Speech-to-Text offers support for Punjabi, Dialogflow does not. When Punjabi is selected, the request is translated into Hindi using Google Translate API and sent to Dialogflow. The agent then services the request in Hindi as before and returns a response which is translated back into Punjabi and displayed to the user. Accompanying Punjabi text is a spoken response in Hindi provided by Google's Text-to-Speech API. Despite the multiple network round-trips required to service requests in Punjabi, negligible additional latency was experienced and responses times did not drop.

In practice, spoken Hindi and spoken Punjabi are very similar and often spoken in neighbouring states (see Figure 5.12). While Punjabi is the official language of Punjab, Haryana and Delhi, Hindi is the official language of neighbouring Himachal Pradesh, Uttar Pradesh, Haryana and Rajasthan among other non-neighbouring states [113]. The close similarity between the languages made this method an acceptable compromise given the technical constraint. As a Punjabi speaker with no previous knowledge of Hindi, I was personally able to understand responses well despite the linguistic differences.

Transliterations

Anecdotal observation of Indian WhatsApp users revealed the prevalence of transliterations in communication. Hindi and Punjabi is often written in Latin script with words being interchanged within a sentence or between sentences. The BCP-47 codes for these hybrid languages are hi-Latn and pa-Latn [114]. To accommodate a wide range of literate ability and linguistic preferences, functionality was added to allow users to communicate in these languages. To illustrate, words such as "Atta", "Chai", "Aloo" meaning "Flour", "Tea" and "Potato" were integrated allowing the user to freely and naturally converse.

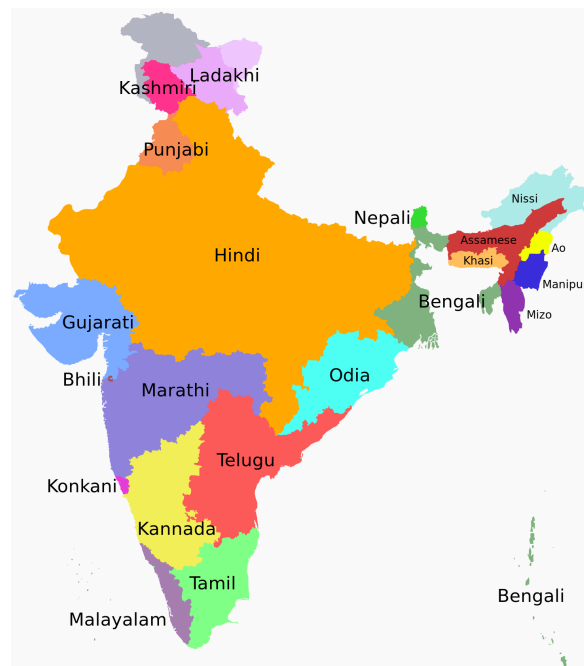


Figure 5.12: Languages in India [113]

5.2.7 Languages

In addition to English, the app offers the ability to switch languages to Hindi or Punjabi by selecting an option in the top app bar. There are two types of translations. String resources and their translations provided in localised versions of `strings.xml` are automatically set by the device according to the locale. These strings can be thought of as static and cover parts of the user interface independent from the database. They include the top app bar, bottom navigation view, chatbot welcome message and other parts of the UI. On the other hand, data retrieved from the database in English is translated into the desired language by accessing its mapping held in a bidirectional `HashMap`. By extending the standard Java `HashMap` to retrieve keys from values, I augmented its functionality to allow two-way translation. A one-to-one unique mapping between keys and values in all three languages was guaranteed since linguistic scripts are unique. The `BiMap` contains all possible crops, markets, states and miscellaneous strings. The Google Translate API was used to generate translations in Hindi and Punjabi which were manually corrected and verified.

Rather than provide separate localised versions of the application for different languages, which are determined on installation from the Google Play Store, users are given control over language with an in-app drop-down menu. This design feature was informed by recent research that investigated language preferences in smart-phone interface design. Karusala et al. found that despite a lack of English fluency, users “frequently engage in English communication proactively and enthusiastically” [115]. Providing control over language options caters to low-literate and non English-speaking users’ aspirations for English fluency. In their study, the au-

thors found participants’ affinity for English came from their desire for social mobility and associated it with topics such as “impressing someone, communicating with authority, and in general, getting ahead in life” [115]. Kentaro Toyama proposes people’s aspirations “provide a better framework than people’s needs when designing to support positive social change” and that design for development must support users goals and aspirations [116]. Aspirations-based design is evidently complementary with Alan Cooper’s Goal-Directed Design [66].

```

1 public class BiMap<K, V> extends HashMap<K, V> {
2     Map<V, K> reverseMap = new HashMap<V, K>();
3
4     @Nullable
5     @Override
6     public V put(K key, V value) {
7         reverseMap.put(value, key);
8         return super.put(key, value);
9     }
10
11     public K getKey(V value) {
12         return reverseMap.get(value);
13     }
14 }

```

Listing 5.1: BiMap.java

The same technique is implemented in the Dialogflow agent’s fulfillment logic. Crops, markets and miscellaneous strings are retrieved from the database in English and translated to Hindi when the device language is set to Hindi or Punjabi.

```

1 class BiMap {
2     constructor(map) {
3         this.map = map;
4         this.reverseMap = {};
5
6         for (const key in map) {
7             const value = map[key];
8             this.reverseMap[value] = key;
9         }
10    }
11
12    toEnglish(key) {
13        return this.map[key];
14    }
15
16    toHindi(key) {
17        return this.reverseMap[key];
18    }
19 }

```

Listing 5.2: BiMap.js

5.2.8 Text-to-Speech

Design for low-literate users motivated the use of speech-based interfaces. Users can tap on a crop card to hear prices spoken aloud. Cards in `CropFragment` were designed with a slight elevation from the surface to invite interaction. When comparing prices in `CompareFragment`, users can similarly tap on a list item and hear tab headings spoken aloud as they navigate between them.

The title and subtitle positioned in the app's toolbar can also be clicked to hear aloud. Text-to-Speech announces the market at which prices are delivered and the date of last update. In `CompareFragment` the crop name takes the place of the subtitle. In this way, the user is always aware of their position within the app. The toolbar provides a familiar signpost that the user can always return to when locating oneself.

Initial testing revealed significant latency when initialising Text-to-Speech upon launch. Since this procedure began only when the user clicked on a expressible item, users would often repeatedly click while the app remained unresponsive for several seconds. Initialization of the Text-to-Speech service was moved to the `MainActivity`'s `OnCreate` method which executes immediately after the app is launched. This solution significantly reduced delay to <1 second for a more seamless user experience.

5.3 Price Forecasting

Forecasted prices help users make informed decisions in the presence of price volatility in agricultural markets. By anticipating future prices, users can optimally decide when and where to conduct their purchases and sales. A coloured arrow icon was added to each crop card with the next-day price prediction.

5.3.1 Long Short-Term Memory Models

Price forecasts were achieved with the implementation of long short-term memory (LSTM) models. These models were introduced in 1997 as a type of recurrent neural network (RNN) that are particularly well-suited for sequence prediction problems deriving from their ability to identify and preserve long-term temporal dependencies within data [117]. They are widely applied in machine translation and natural language processing applications due to their ability to understand semantic and syntactic context [118]. Examples include Apple's QuickType word prediction keyboard [119] and Facebook's automatic language translations [120]. LSTMs are also commonly applied in price forecasting with long sequences of time series data. They were originally developed to address the problem of vanishing gradients that can occur during the backpropagation phase of training traditional RNNs [121]. In one study, LSTMs were shown to offer superior predictive performance on univariate time series data relative to traditional autoregressive models, such as an ARIMA model [122]. The authors cite average reductions in error rates from ARIMA models

between 84 and 87 percent [122].

The fundamental computational units of long short-term memory models are memory cells (see Figure 5.13). They differ from traditional recurrent neural networks in that a cell's state is regulated by a constant error carousel (CEC) that enables the uninhibited flow of information over long time lags [121]. An input, output and forget gate regulate the addition and removal of information from the cell's state [123]. Together, the repeated modules in an LSTM enable it to preserve information over thousands of time-steps [118].

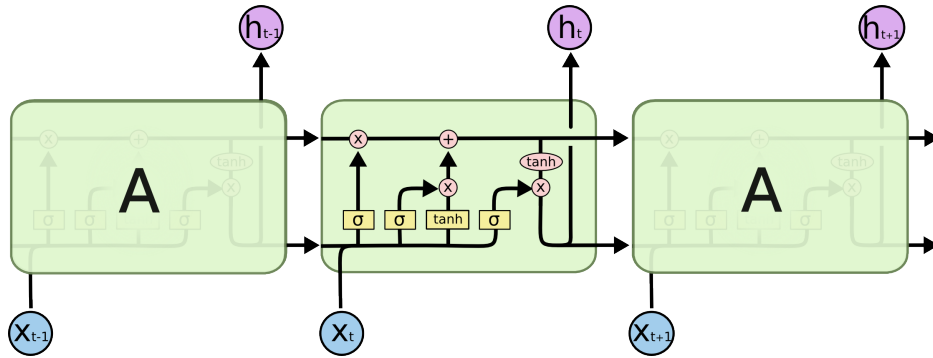


Figure 5.13: LSTM Structure [118]

5.3.2 Model Setup

LSTM models were built with the Tensorflow 2 Keras API in Python [124]. Two stateful LSTM hidden layers, each with 5 neurons (memory cells), are stacked and joined to a fully connected Dense output layer with a single output neuron. The output layer contains a linear activation function that predicts the crop price value in the next step, since this is a regression-type problem (as opposed to classification). This problem was framed as a many-to-one sequence prediction problem, as evidenced by the 5 input time steps and 1 output value. I arrived at this model architecture after a process of trial and error tuning hyperparameters.

Data Preprocessing

Ten years of daily prices of twenty-two crops at over one hundred markets was sourced from India's National Informatics Centre [82]. Over 3000 daily reports were merged into one complete csv file and preprocessed to prepare for model training. The index was converted to datetime format and summary statistics were removed. Tables were then split by location with each market having its own csv.

In 2010, only 30 markets reported crop prices. Since LSTMs require large amounts of data for training, it was infeasible to forecast accurate predictions for newer markets where less than a year of data was present. With this limitation, models were developed for the remaining 30 markets across 22 crops. In total, 660 unique models were created.

Model Training

Each model is trained on roughly 90% of the dataset (3030 observations) and tested on 10% (365 observations). Since crop prices were observed to generally increase over time, the series is defined as non-stationary. For more skillful forecasts, systematic trends and seasonality were removed from the data by taking the difference between the values at the current (t) and previous time period ($t - 1$). This leaves us with a series of differences over time. The series is then normalised to within $[-1, 1]$ as required by the LSTM's *tanh* activation function. As a regression problem, the mean squared error was chosen as the loss function. The adam optimiser was chosen due to its optimal performance and efficiency. A process of trial and error revealed loss plateaued after 5 epochs. For greater efficiency and to prevent over-fitting, training was halted at this point. One study suggests that the number of epochs selected while training an LSTM model has “no effect on the performance on the trained forecast model” when applied to the prediction of stock prices [122].

5.3.3 Results

Root mean squared errors (RMSEs) were calculated after performing a walk-forward validation on the test dataset. RMSEs are easily interpreted as they are defined in the same units as the forecast data. Over the entire set of predictions, average RMSE was 1.43. In other words, the mean error between price forecasts and actual observed data was ± 1.43 Indian rupees, averaged across all markets and crops. Minimum and maximum RMSE was 0 and 30 meaning the best forecast correctly predicted prices and the worst forecast was inaccurate to ± 30 Indian rupees. Accuracy was additionally calculated as the absolute difference between the prediction and actual value, as a percentage of the actual value. Averaged across all markets and crops, mean accuracy was 97.9%. Minimum and maximum accuracy was 85% and 100%.

Evaluation

However, when compared with the baseline, RMSEs show that LSTM price forecasts are approximately equivalent to those forecasted by a persistence model. A persistence model forecast is one where the observation at the current time step is used to predict the next. Figures 5.15(a) and 5.15(b) show that the RMSEs between the LSTM model and the persistence model are almost identical. Average RMSEs for the LSTM model and persistence model were 1.432 and 1.426 respectively. Interesting to note is the lower predictive accuracy for potato, onion and tomato (see Figure 5.14). Since these crops are not covered under the minimum support price (MSP) regulation, these crops' prices experience greater volatility negatively impacting the models' forecasting accuracy. As vegetables, these crops are categorically different to the other grains, pulses and oilseeds. Recent research confirms high short-term price volatility among these three crops in particular, and fruits and vegetables in general due to demand-side factors [125].

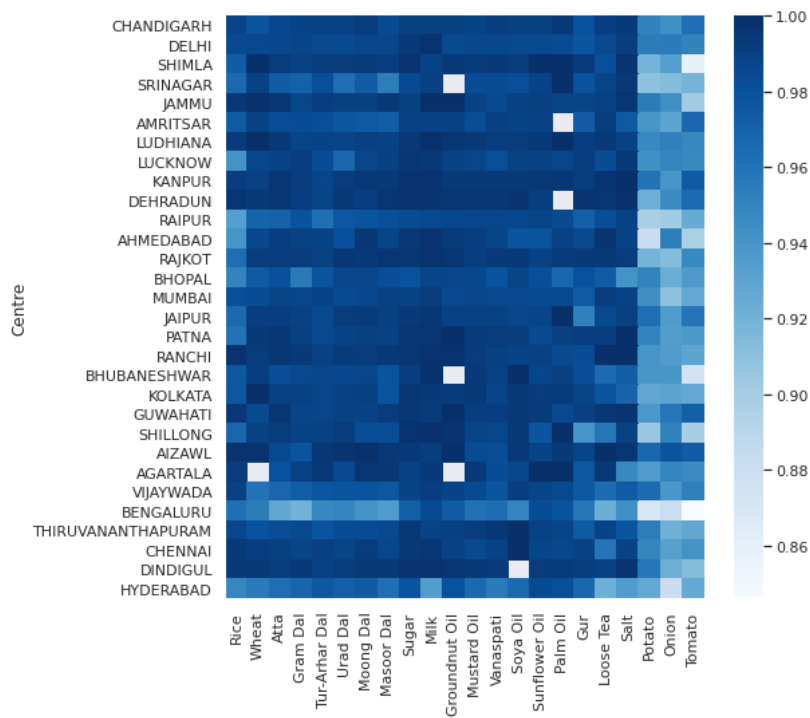


Figure 5.14: Accuracy Heatmap of LSTM Model

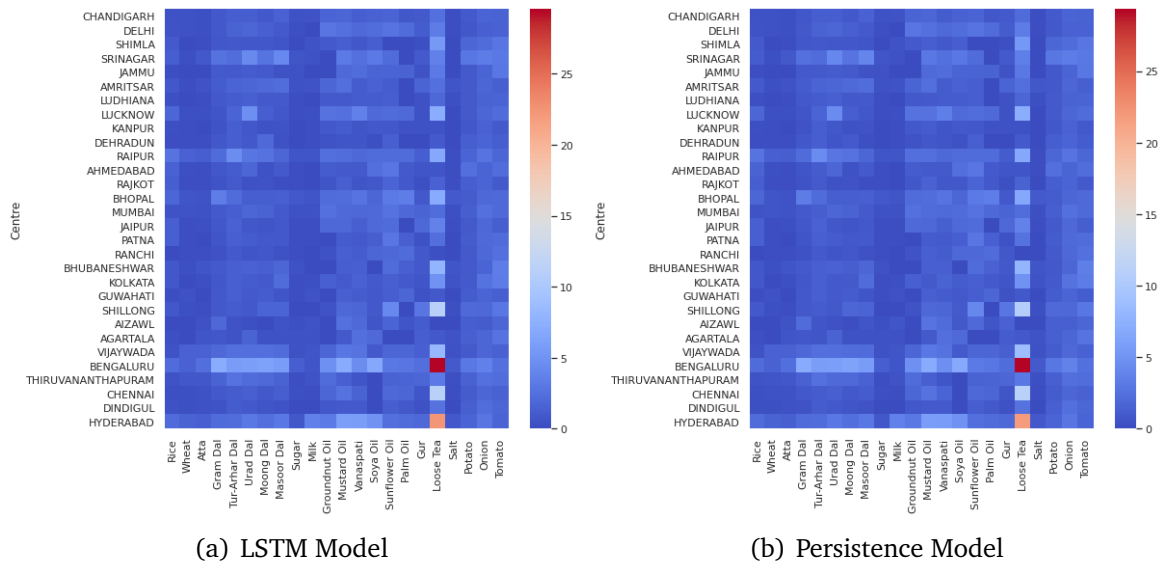


Figure 5.15: RMSE Heatmaps

Chapter 6

Evaluation

The evaluation of user experience necessitated thorough usability testing. Using the best practices outlined in Carol Barnum's Usability Testing Essentials [126], both formative and summative testing were incorporated into fast and focused iterative development cycles. In her book, Barnum describes usability as, defined by the International Organisation for Standardisation (ISO), "The extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.". This concept of usability is highly complementary with Alan Cooper's own Goal-Directed Design in which users' goals, not tasks, are prioritised and facilitated by the careful design of interaction [66]. Whether its Cooper's form, content, and behaviour design concerns [66], Norman's reflective, behavioural, and visceral processing levels [65] or Barnum's efficiency, effectiveness and satisfaction measures of usability [126], each uniquely contribute insights to the design and evaluation of user experience.

6.1 Heuristics Evaluation

A heuristics evaluation is typically used to inspect an interface using a specific set of guidelines in search of usability violations [126]. Figure 6.1 demonstrates how particular features of Mobile Mandi were designed to meet the usability heuristics for user interface design as proposed by Jakob Nielsen [127]. These are:

1. Visibility of system status

"The system should always keep users informed about what is going on, through appropriate feedback within reasonable time."

When the app launches, a progress bar is displayed while information is being retrieved from the database.

2. Match between system and the real world

"The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms."

Follow real-world conventions, making information appear in a natural and logical order”.

Crop images provide a visual metaphor for the action of visiting an Indian market. Crops always appear in the same order helping the user locate information of need through repeated interaction.

3. User control and freedom

“Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Supports undo and redo and a clear way to navigate”.

While scrolling, the bottom navigation bar and toolbar do not disappear which constantly provides the user with a way to return to the home page. Ability to favourite markets gives the user control over their experience with the app.

4. Consistency and standards

“Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions”.

Icons and label choices were informed by global conventions, such as the help, speaker, favourite and translate icons. Teal-coloured (secondary colour) floating action buttons are used to nudge users towards performing specific actions, such as the search button and microphone button. Purple-coloured (primary colour) buttons are used to indicate intent-passing between pages and to external apps.

5. Error prevention

“Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action”.

The app was designed to gracefully handle permission rejections. For example, if the user denies locations permissions, modal prices are displayed instead of displaying a blank screen. Try-catch statements were used extensively to prevent surfacing exceptions and provide the user with an error-free experience.

6. Recognition rather than recall

“Minimize the user’s memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate”.

Toolbars are never hidden from view and remain always visible to the user. Conventions are followed in the selection of icons and text to leverage users' existing mental models from technology use. When comparing crops, the crop name is added as a subtitle in the toolbar to avoid relying on the user's memory of which crop was compared.

7. Flexibility and efficiency of use

“Accelerators – unseen by the novice user – may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions”.

When location permissions are granted, the app automatically detects the closest market to the user. This information is saved and updated if users' coordinates change. Markets are automatically ordered by closest market first, presenting the user with a list of likely candidates for markets of interest.

8. Aesthetic and minimalist design

“Dialogues should not contain information, which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. Visual layout should respect the principles of contrast, repetition, alignment, and proximity”.

The linear gauge chart is presented with minimal text to avoid overwhelming and impeding the user's ability to scan the page. Text is removed from the bottom navigation bar while not in focus. Prices appear large in bold as they provide the most important information.

9. Help users recognize, diagnose, and recover from errors

“Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution”.

The chatbot's error messages provide the user with helpful hints about how to solve the error.

10. Help and documentation

“Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large”.

The help dialog provides additional information without cluttering the main view.

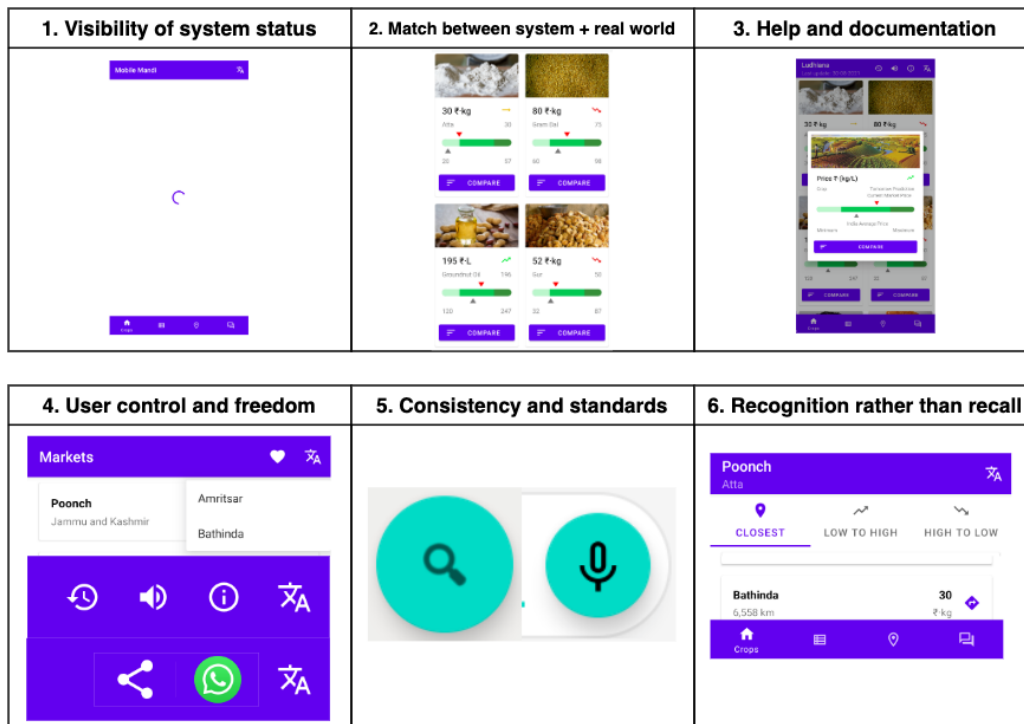


Figure 6.1: Heuristics Evaluation

This set of 10 heuristics are efficiently encapsulated by Whitney Quesenberry's 5Es, "effective, efficient, engaging, error tolerant and easy to learn" [128]. These five dimensions of usability were evaluated heuristically and formally, as demonstrated in the next section.

6.2 Usability Testing

6.2.1 Setup

A usability study was conducted with three participants from Kapurthala (a city in the state of Punjab) and its surrounding rural areas. This non-random selection of participants were recruited using the snowball type of purposive sampling where initial participants nominated additional eligible participants [129]. Snowball sampling is a standard technique to engage hard-to-reach communities for the purposes of research [130]. Semi-structured interviews were around 40 minutes long and took place in the participant's home in their native language (Punjabi) using video conferencing software. Scenario testing with three additional participants fluent in English but illiterate in Hindi was used as an alternative to primary interviews with low-literate users. In these scenarios, the app's language was set to Hindi to crudely approximate the experience of a low-literate user, although this approach has its limitations (discussed later). Participant names were anonymised for use in this paper. All participants were reported to have at least five years of experience using smartphones.

Demographic Information

Participant	Gender	Age Range	Location	Education	English— —Hindi— —Punjabi Fluency
Aman	M	30 - 40	Kapurthala	High school	I — A — A
Samir	M	30 - 40	Kapurthala	High school	I — I — A
Kamal	M	40 - 50	Kapurthala	College	I — A — A
Simran	F	18 - 25	United Kingdom	High school	A — N — B
Param	M	18 - 25	United Kingdom	College	A — N — B
Jasleen	F	18 - 25	United Kingdom	High school	A — N — B

Table 6.1: Participant Demographics;
N = None, B = Basic, I = Intermediate, A = Advanced

Participants were given a pre-test questionnaire to ensure demographic information appropriately matched those of the ideal user group (see Table 6.1). They were then told of the purpose and structure of the study. Users were asked to complete a series of tasks with the app while thinking aloud. These tasks were structured as scenarios which tested the app's effectiveness and efficiency in enabling the user to accomplish their goals. The 'thinking aloud' technique was leveraged to elicit less biased responses given the well-documented acquiescence bias [131]. A moderator's script, lightly adapted from [126], was used to ensure participants were not biased by variations in instructions. The mobile app analytics platform UXCam [132] was installed in-app to provide detailed analysis of user experience. A heatmap (see Figure 6.2) was generated from user's interactions with the app to understand interaction patterns.

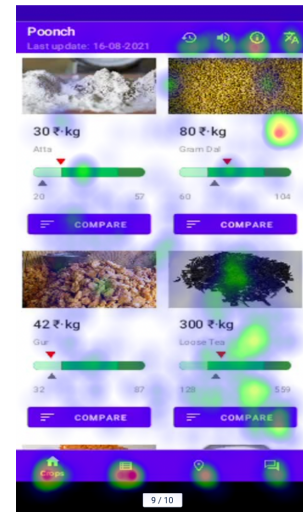


Figure 6.2:
Interaction Heatmap

Three standard post-test questionnaires were given to assess user experience and satisfaction. Standard questionnaires were used for their research-backed formulation and validation as well as the ability to benchmark across previous findings. The System Usability Scale (SUS) [133] and Post-Study System Usability Questionnaire (PSSUQ) [134] use 5-point and 7-point Likert scales ranging from "Strongly Agree" to "Strongly Disagree". The User Experience Questionnaire (UEQ) [135] which consists of a 7-point scale between pairs of opposing adjectives was also used to measure perceived usability.

6.2.2 Results

System Usability Scale

The System Usability Scale (see Appendix A) developed in the mid-1980s consists of 10 questions with responses ranging from 1 (Strongly Disagree) to 5 (Strongly Agree). Despite its self-description as a “quick and dirty usability scale”, analysis from 10 years of data certified the tool as “highly robust and versatile” for assessing usability [136]. The mean average from all participant surveys resulted in a score of 92.5, exceeding the mean of 75.24 for OS-based graphical user interfaces (GUIs) [136]. Among SUS mean ratings for everyday products, the app’s usability ranks above web browsers (88.1) but below Google search (93.4) [137].

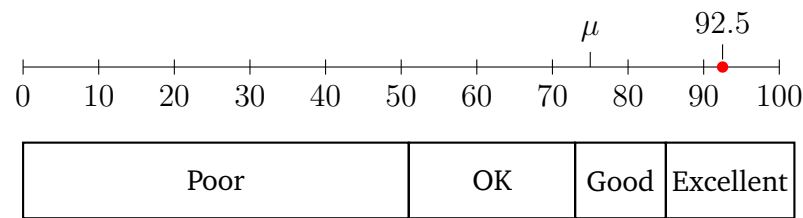


Figure 6.3: System Usability Scale

Post-Study System Usability Questionnaire

The PSSUQ (see Appendix B) was designed by IBM in the mid-90s to assess users’ perceived satisfaction with computer systems [134]. The questionnaire consists of 19 questions assessed on a scale of 1 (Strongly Agree) to 7 (Strongly Disagree). It was designed to evaluate a product’s performance across a set of 5 usability characteristics:

- 1. Quick completion of work
- 2. Ease of learning
- 3. High-quality documentation
- 4. Functional adequacy
- 5. Rapid acquisition of productivity

Once results are gathered, scores across four dimensions (Overall, System Usefulness, Information Quality and Interface Quality) are calculated by computing the means of sets of response values [137]. Lower scores indicate higher satisfaction. Analysis from 5 years of lab-based usability studies found the reliability scores, also known as Cronbach’s Alpha, ranged between 0.83 and 0.96, [138] consistent with earlier estimates which exceeded 0.85 [134]. Coefficient alpha is measured on a scale from 0 (perfect unreliability) to 1 (perfect reliability) [136]. Empirical evidence supports the validity of the PSSUQ as a standard usability measure. Figure 6.4 shows the results of the questionnaire, with scores all exceeding the means from

a dataset of 21 studies and 210 participants [137].

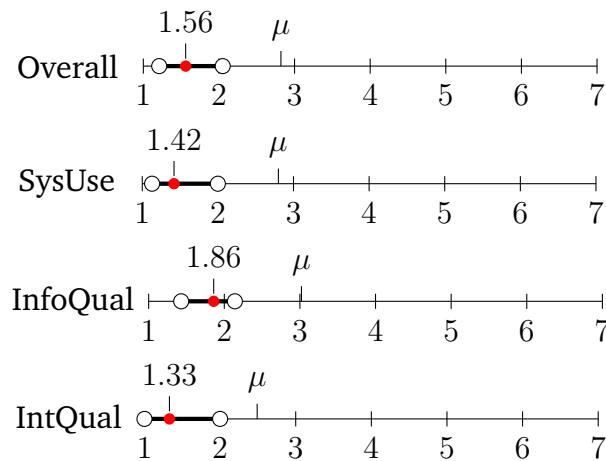


Figure 6.4: Post-Study System Usability Questionnaire Results

User Experience Questionnaire

The User Experience Questionnaire [135] was designed in 2005 by SAP to measure user experience across 6 scales:

1. Attractiveness – “What is the user’s overall impression of the product?”
2. Perspicuity – “Is it easy to learn how to use the product?”
3. Efficiency – “Can users solve their tasks without unnecessary effort?”
4. Dependability – “Does the user feel in control of the product?”
5. Stimulation – “Is it exciting and motivating to use the product?”
6. Novelty – “Is the product innovative and creative?”

The questionnaire aims to quantify the more nebulous aspects of interaction design that were previously absent in traditional usability measures, such as user experience goals. Across the six dimensions, it unifies the assessment of both pragmatic (goal-directed) and hedonic (non-goal-directed) qualities of a system [135]. Results from the questionnaire found mean scores across all six scales were greater than or equal to the 90th percentile in a dataset of 21175 participants from 468 studies [139] (see Table 6.2).

Scale	Mean	Standard Deviation	Benchmark Comparison	Interpretation
Attractiveness	2.78	0.06	Excellent	\geq 90th Percentile
Perspicuity	3.00	0.00	Excellent	\geq 90th Percentile
Efficiency	2.83	0.02	Excellent	\geq 90th Percentile
Dependability	2.58	0.15	Excellent	\geq 90th Percentile
Stimulation	2.00	0.81	Excellent	\geq 90th Percentile
Novelty	1.92	0.58	Excellent	\geq 90th Percentile

Table 6.2: User Experience Questionnaire Results

Product Reaction Cards

A set of 118 product reaction cards were developed in 2002 by usability professionals at Microsoft to measure the similarly intangible aspect of desirability in a user's experience with a product [140]. Designed to avoid the acquiescence bias that formal usability questionnaires are prone to, there is an approximate split of 60% positive words and 40% negative words. Acquiescence bias is the tendency for test participants to give higher-than-average positive responses when completing post-test questionnaires [126] since more people are likely to agree with a statement than to disagree [137]. Consequently a mixture of positive and negative words helps to counter this phenomenon.

Participants were asked to select 3 to 5 words to summarise their experience of using the app and how it made them feel. "Accessible" and "Easy to use" are among the words and phrases that appeared in higher frequencies. To visualise users' impressions, a word cloud was generated with these findings with frequently chosen words appearing larger (see Figure 6.5).



Figure 6.5: Product Reaction Word Cloud

6.3 Stress Testing

In addition to usability testing, Robo testing was employed to uncover stability and performance errors. Integrated with the Google Play Console, Firebase Test Lab enabled custom automated testing across a wide range of Android devices and operating system versions [141]. Robo test explores the app simulating real user interaction by navigating through pages, performing click events, typing and scrolling. At the end of a test, logs, screenshots and videos are displayed in the Play Console in the form of a pre-launch report. All major stability, performance and accessibility issues, such as null pointer exceptions, slow frame renderings and small touch targets, were resolved before successfully publishing the app to the Google Play Store.

Stress testing was performed by the UI/Application Exerciser Monkey [142]. Generally known as a ‘chaos monkey’, it is a command-line tool running on the Android Debug Bridge (ADB) that sends a pseudo-random stream of events to the system and reports if the system has crashed or is unresponsive [142]. Stress testing results from a study of popular Android apps (e.g., Facebook, WhatsApp, Snapchat) found the mean time to crash was 85 seconds and 7,513 events [143]. Repeated testing found the app was stable and responsive to at least 10 thousand events.

```
Last login: Mon Aug 23 17:02:42 on ttys000
jaskirat@Jaskirats-MacBook-Pro ~ % adb shell monkey -p
↳ com.jaskiratchahal.firebaseio -v 10000
:Sending Touch (ACTION_DOWN): 0:(41.0,618.0)
:Sending Touch (ACTION_UP): 0:(73.24594,690.2041)
:Sending Trackball (ACTION_MOVE): 0:(1.0,1.0)
...
Events injected: 10000
:Sending rotation degree=0, persist=false
:Dropped: keys=0 pointers=0 trackballs=0 flips=0 rotations=0
## Network stats: elapsed time=71338ms (0ms mobile, 0ms wifi,
↳ 71338ms not connected)
// Monkey finished
jaskirat@Jaskirats-MacBook-Pro ~ %
```

6.4 Android Studio Profiler

Design for mobile devices requires careful consideration of system resource consumption. These concerns are further augmented when designing for the developing world. CPU, memory, network and battery usage were measured and opti-

misused with the monitoring tools integrated in Android Studio [144]. Where possible, optimisations were made to conserve mobile resources, such as replacing `ValueEventListeners` with `SingleValueEventListeners`. Unnecessary network calls were removed by combining multiple atomic operations into one. Location requests were set with `PRIORITY_LOW_POWER` which retrieves the device's location with coarse accuracy (to about 10km) and consumes less power [145]. City-level location accuracy is ideal since the application does not yet widely support intra-city markets but can be modified in the future to balance priorities between power and accuracy (to 100m). Memory leaks were checked and dormant resources released on termination. App cache size was tracked during prolonged use and was verified to remain stable.

6.5 Google Play Store

In the process of publishing to the Google Play Store, a selection of promotional materials were prepared. A simple app icon was designed with a white tractor icon against a green background. Screenshots were designed to promote key features and capabilities.

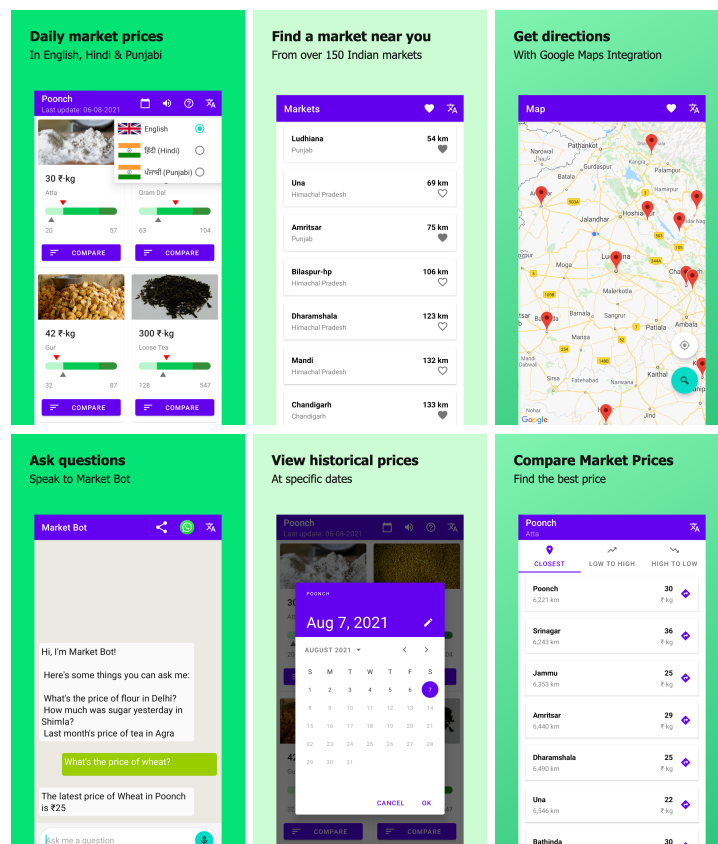



Figure 6.6: Promotional Screenshots



Mobile Mandi

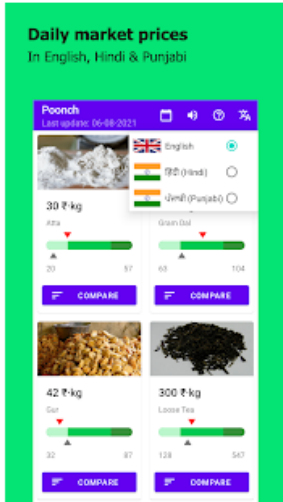
Jaskirat Chahal Business

3 PEGI 3

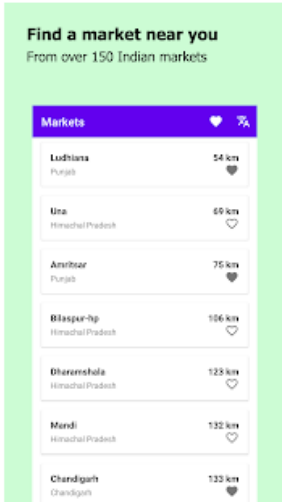
This app is available for all of your devices

Installed

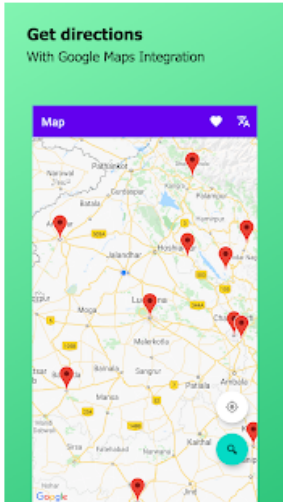
Daily market prices
In English, Hindi & Punjabi



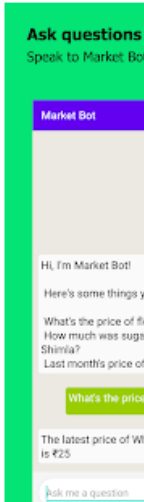
Find a market near you
From over 150 Indian markets



Get directions
With Google Maps Integration



Ask questions
Speak to Market Bot



Mobile Mandi presents daily prices for 22 essential crops across 157 markets across India. You can also:

- listen to market updates
- speak with a chatbot
- compare prices between markets
- view historical data
- get directions and
- share information with friends on WhatsApp.

Languages supported include: Hindi, Punjabi and English.

Figure 6.7: Store Listing

Chapter 7

Conclusion

Indian farmers and consumers lack access to real-time market information which results in poor market outcomes and inefficient equilibria. Improving access to information can potentially eliminate market asymmetries and improve the efficient functioning of physical agricultural markets. Indeed previous work supports this conclusion showing similar interventions increasing modal prices of agricultural commodities [71].

Mobile Mandi improves the price discovery mechanism by providing localised, actionable, accurate and timely market information in an accessible format designed for low-literate users from the ground-up. Following the principles and patterns of interaction design, the app closely represents the users' mental models by imitating the real world. The goal-directed design approach ensured a focus on helping users achieve their goals with efficiency, effectiveness and satisfaction.

Following a review of the literature, personas were constructed to guide human-centred design decisions. From these research-backed fictional accounts, a set of design requirements were identified. Mainstream interaction flows were studied to provide a familiar interface while minimising cognitive effort. Strategic use of a single-activity architecture was leveraged to provide a seamless user experience. Wireframes were designed with minimal hierarchical levels. A navigation diagram models the flow in and out of the app.

Price data on 22 crops at 157 markets was sourced from India's National Informatics Centre. A web scraper was built, scheduled and automated on a virtual machine that retrieves data daily. It was designed to ignore unreported prices and always provide the user with the latest information. Firebase Realtime Database was selected due to its synergies with Android and offline capabilities. Google Geocoding API provided market coordinates to present the user with their closest markets.

On launch, the user immediately sees prices from their closest market. The app automatically detects the user's location which eliminates unnecessary setup. Location services were optimised to conserve mobile resources. Each device is anonymously authenticated with Firebase which facilitates the storage of user preferences. The

user can query historical prices at any date and market, listen to market prices, get help, translate, and compare prices with one-click. A custom linear gauge chart was designed to show prices in context. Google Translate API provided translations in Hindi and Punjabi which were manually verified. Google Maps API provides mapping functionality. An intelligent virtual assistant was created with Dialogflow in Node.js to provide an intuitive speech-based interface. The assistant was integrated with WhatsApp by connecting the Twilio API with Dialogflow. A Docker container enclosing the Express.js web-service was deployed to Google Cloud Run. The server was modified to handle simultaneous requests in multiple languages. Each message shared to/from WhatsApp contains a Deep App Link that redirects the user to the app or to the Google Play Store if not installed.

Next-day price forecasts were generated by training stacked long short-term memory (LSTM) models on a univariate time series dataset of daily crop prices over ten years. Root mean squared error was found to be 1.43, averaged over markets and crops. Mean accuracy was 97.9%. Despite good performance, LSTMs did not offer superior performance than the baseline.

Evaluation comprised heuristic evaluation, usability testing and stress testing. The app was demonstrated to meet several dimensions of usability. Usability testing with Punjabi farmers quantified the quality of user experience, with scores above average in all criteria. Scenario testing was carried out in the absence of low-literate users. Standardised questionnaires were used for their strong validity and enabled performance benchmarking. Testing with Firebase Test Lab verified the absence of stability, performance, accessibility, and security flaws. Stress testing with the Application Exerciser Monkey revealed above average performance among popular apps.

The app was successfully launched to the Google Play Store. Mobile Mandi demonstrably meets and exceeds the design requirements previously outlined, with above average performance. The next section discusses limitations.

7.1 Limitations

Though objectives were met, there exist limitations. First, the limitations on data restrict the commercial viability of the application for Indian users. Crop price data was sourced from India's National Informatics Centre on 22 crops at 157 markets. However, there are over 27,000 wholesale and primary markets in India [2]. Long distances between markets make it infeasible for farmers to act on market information due to the difficulties in storing and transporting perishable crops. Furthermore, there are over 100 crops grown and harvested in autumn and winter alone [146]. More crops and greater density of markets is required to increase the utility of the application for its users.

The methods of usability testing also carry some limitations. Since low-literate users could not be sourced, scenario testing with the in-app language set to one not recog-

nised by the participant was used to approximate the experience of a low-literate user. However, research shows that beyond literacy, mental models and cognitive abilities of literate and non-literate users differ significantly. One study noted users differed in their visual organisation, visual memory, mental spatial orientation, and speed of cognitive processing among others [53]. These cognitive differences in low-literate users affect interaction and could not be accounted for. Therefore further testing is required to verify ease-of-use among low-literate users.

Due to hardware limitations it was not possible to test the app (via emulation in Android Studio) on the oldest versions of Android OS. Though test participants and stress testing verified stable performance on a variety of devices and OS versions, there exists the possibility of incompatibility in older devices despite measures being taken to ensure backwards compatibility.

Also discussed briefly in a previous chapter was the accuracy of price forecasts. Despite the model achieving average predictive accuracy (defined by the root mean squared error) of ± 1.43 Indian rupees, it was not found to improve significantly over the benchmark baseline model. Further work with multivariate, deeper and/or more complex architectures may be explored to increase predictive accuracy.

7.2 Ethical and Professional Considerations

This section addresses a number of ethical and professional considerations. The first concern originates from the collection and storage of data on the application's users. In compliance with the UK's Information Commissioner's Office (ICO) "Guide to Privacy and Electronic Communications Regulations" [147] all location data is anonymised and its collection consented to when the app is initially opened after installation. Following app permissions best practices [148], users may freely deny permission without significant loss of functionality. In this case localised data is replaced with average data which is then displayed to the user. The user may manually select the market at which they wish to see information. Users who give consent to the location request have market distances stored in the Firebase Realtime Database under an anonymous unique user identifier. User IDs consist of an arbitrary sequence of characters that persist for the duration the app is installed. Database contents may only be accessed by authenticated users with the proper credentials. Furthermore, the database is secured under several layers of encryption protected by the Advanced Encryption Standard (AES) algorithm AES-256 as recommended by the US National Institute of Standards and Technology (NIST) [149]. According to the ICO, "there is presently no known practical attack that could brute-force an AES 256 key" [147]. In addition, messages sent to Dialogflow during interactions with the virtual assistant are similarly encrypted at rest [149] and in transit [150]. Alternatively, if the user wishes to communicate via speech, microphone permission is first requested and, if granted, similarly encrypted in transit.

The second concern relates to the collection and processing of data on test participants in the usability study. Demographic information was collected after consent was requested and given. These include: gender, age range, city-level location, education level, language fluency and years of smartphone use (see Table 6.1). Participant names were anonymised for use in this paper. Furthermore, user interaction was logged with the mobile app analytics platform UXCam [132] for the duration of usability testing. Information collected included: device type, OS version, city-level location, number of sessions, engagement time, and gestures. This data was stored by UXCam with Amazon Web Services and encrypted at rest using the AES-256 encryption algorithm [151]. Any personally identifiable information was not recorded. Interaction data is no longer being logged and data has since been deleted completely.

The third concern relates to the provision of next-day price forecasts. Despite good predictive accuracy, potential monetary loss from the reliance on price forecasts is of material ethical concern. To address this concern, an in-app dialogue displays a disclaimer on launch.

The next chapter presents opportunities for further enhancement.

Chapter 8

Future Work

Further development has the potential to increase the commercial utility of the application. A number of key improvements are described in this chapter.

8.1 Data

A key takeaway from the testing phase of the application was the importance of data. Test participants noted that the selection of crops and markets was limited, and that they would like to see both retail and wholesale prices.

8.1.1 Crops

The application provides users with prices of 22 essential crops. 10 of these are covered under the Indian government's Minimum Support Price regulation. These are:

- | | |
|-------------------|-----------------|
| • Paddy (Rice) | • Urad |
| • Wheat | • Peanut oil |
| • Chickpea (Gram) | • Soyabean oil |
| • Tur (Arhar) | • Sunflower oil |
| • Moong | • Sugarcane |

The 12 remaining crops reported in the data are not covered by the regulation and so prices are open to market competition. Farmers commented that these non-subsidised crops, in addition to fruits and vegetables, were of a higher priority due to greater price volatility. To fulfill these needs, a reliable data source must first be discovered. However there is difficulty in finding publicly available market information as Indian market operators typically keep physical paper records.

In addition to greater data on agricultural crops, farmers expressed an interest in prices of fertilizers and pesticides. Data could also be extended to non-agricultural commodities such as silver and gold as they are popular stores of value in India.

8.1.2 Markets

During testing, farmers indicated the need for greater geographic market density. Since prices can vary considerably between markets within a city, it is important to provide more geographically relevant data. For example, test participants in Kapurthala, Punjab were limited to market prices in Amritsar (70km) or Ludhiana (100km).

8.1.3 Prices

Complementary to retail prices, wholesale prices may also be included to cater to large-scale purchasers and consumers. The data source used in this project additionally reports wholesale prices in Indian rupees per quintal [82]. This marginal improvement helps the application to cater toward non-retail users.

Crowd-sourced prices are another novel idea for exploration. Although it was decided to prevent users from self-reporting prices in the application, further work could be done to design a mechanism for reporting prices robust to misinformation. Such an idea might draw inspiration from the design of Waze, a popular mobile navigation app that allows drivers to alert fellow drivers of traffic, accidents and hazards [152]. Users of the app may up-vote or down-vote posts to validate or invalidate the authenticity of the report.

8.2 Languages

In addition to English, Hindi and Punjabi, further work might extend support to a greater number of Indian dialects and languages. Difficulty lies in supporting India's abundant dialects and low-resource languages. Recent work has leveraged radio archives to develop an novel intelligent assistant for illiterate speakers of West African languages: Maninka, Pular, and Susu [55].

8.3 Price Forecasts

Price forecasts generated with long short-term memory (LSTM) models demonstrated good performance in predicting future observations from a univariate time series. However, performance did not meaningfully improve on the performance of a persistence model. Further improvements to the hyperparameters of the model may be explored. The model may also be augmented with additional data on market supply, rainfall, temperature, humidity, groundwater levels and other determinants of market price to improve its predictive accuracy.

Bibliography

- [1] Devesh Kapur and Mekhala Krishnamurthy. Understanding Mandis: Market Towns and the Dynamics of India's Rural and Urban Transformations. *Center For The Advanced Study of India, University of Pennsylvania*, 2014.
- [2] Sriganesh Lokanathan and Harsha De Silva. Leveraging Mobile 2.0 in India for Agricultural Market Access. *SSRN Electronic Journal*, 03 2010. doi: 10.2139/ssrn.1618193.
- [3] Gregory Clark. The Great Escape: The Industrial Revolution in Theory and History. *Davis, UC Davis, Department of Economics*, 2003. URL <http://faculty.econ.ucdavis.edu/faculty/gclark/papers/IR2003.pdf>.
- [4] Thomas Robert Malthus, Niall O'Flaherty, Deborah Valenze, E. A. Wrigley, Kenneth Binmore, and Karen O'Brien. *An Essay on the Principle of Population: The 1803 Edition*. Yale University Press, 2018. ISBN 9780300177411. URL <http://www.jstor.org/stable/j.ctv1bvnf95>.
- [5] Gregory Clark. *A Farewell to Alms: A Brief Economic History of the World*. Princeton University Press, student edition, 2007. ISBN 9780691121352. URL <http://www.jstor.org/stable/j.ctt7srwt>.
- [6] F. Poletti, N. Wheeler, Marco N. Petrovich, N. Baddela, E. N. Fokoua, J. Hayes, D. Gray, Z. Li, R. Slavík, and D. Richardson. Towards high-capacity fibre-optic communications at the speed of light in vacuum. *Nature Photonics*, 7:279–284, 2013.
- [7] World Bank Jobs. Employment in agriculture (% of total employment) (modeled ILO estimate) (SL.AGR.EMPL.ZS), 2019. URL <https://databank.worldbank.org/source/world-development-indicators>. Last accessed on 25/08/2021.
- [8] World Bank World Development Indicators. Agriculture, value added (% of GDP) (NV.AGR.TOTL.ZS), 2020. URL <https://databank.worldbank.org/source/world-development-indicators>. Last accessed on 25/08/2021.
- [9] Ministry of Agriculture and Family Welfare. Agriculture Census 2015-16. All India Report on Number and Area of Operational Holdings, 2019. URL https://agcensus.nic.in/document/agcen1516/T1_ac.2015_16.pdf. Last accessed on 25/08/2021.

-
- [10] The Football Association. The FA Guide To Pitch And Goalpost dimensions, 2012. URL <https://www.thefa.com/-/media/cfa/staffordshirefa/files/facilities-and-funding/the-fa-guide-to-pitch-and-goalpost-dimensions.ashx>. Last accessed on 25/08/2021.
- [11] Rajeev Sharma and Gurpreet Singh. Access to Modern Agricultural Technologies and Farmer Household Welfare: Evidence from India. *International Journal of Asian Studies*, 6:19–43, 03 2015. doi: 10.1177/0976399614563222.
- [12] Thiagu Ranganathan. Farmers' income in India: evidence from secondary data. *Agricultural Situation in India*, 72(3):30–70, 2015.
- [13] BBC News. Have India's farm suicides really declined?, 2014. URL <https://www.bbc.co.uk/news/world-asia-india-28205741>. Last accessed on 25/08/2021.
- [14] National Crime Records Bureau. Accidental Deaths & Suicides in India, 2015. URL https://ncrb.gov.in/sites/default/files/chapter-2A-suicides-in-farming-sector-2015_0.pdf. Last accessed on 25/08/2021.
- [15] Seema Bathla and R. Srinivasulu. Price Transmission and Asymmetry: An Empirical Analysis of Indian Groundnut Seed and Oil Markets. *Indian Journal of Agricultural Economics*, 66(4):1–16, 2011. doi: 10.22004/ag.econ.204784. URL <https://ideas.repec.org/a/ags/inijae/204784.html>.
- [16] Department of Agriculture Ministry of Agriculture and Government of India Co-operation. Final Report of Committee of State Ministers, in-charge of Agriculture Marketing to Promote Reforms, 2013. URL <https://dmi.gov.in/Documents/stminprreform.pdf>. Last accessed on 28/08/2021.
- [17] Ronald Harry Coase. The nature of the firm. *economica*, 4(16):386–405, 1937.
- [18] National Sample Survey Office. Key Indicators of Situation of Agricultural Households in India, 2014. URL http://mospi.nic.in/sites/default/files/publication_reports/KI_70_33_19dec14.pdf. Last accessed on 25/08/2021.
- [19] Press Information Bureau India. Press Release: Digital India, 2021. URL <https://pib.gov.in/PressReleseDetailm.aspx?PRID=1700749>. Last accessed on 28/08/2021.
- [20] Shivani Anand. India's Smartphone Market Registers Double-digit Growth as it Rebounds from a Sluggish H1'20, 2020. URL <https://www.idc.com/getdoc.jsp?containerId=prAP46991220>. Last accessed on 26/08/2021.
- [21] Android Go, 2021. URL <https://www.android.com/versions/go-edition/>. Last accessed on 26/08/2021.
-

- [22] Itel A48, 2021. URL <https://www.flipkart.com/itel-a48-gradation-purple-32-gb/p/itm11a32cb5b94af>. Last accessed on 26/08/2021.
- [23] Labour Department Punjab. Adjustment of Minimum Rates of Wages Punjab, 3 2013. URL <https://pblabour.gov.in/Content/documents/MW%20Punjab%20April.pdf>. Last accessed on 26/08/2021.
- [24] Office of the Labour Commissioner Punjab. Adjustment of Minimum Rates of Wages Punjab, 5 2020. URL <https://www.capsi.in/notifications/The%20Punjab%20Minimum%20Wages%20Notification%201st%20Mar%202020.pdf>. Last accessed on 26/08/2021.
- [25] Smartphone penetration rate in India from 2010 to 2020, 2021. URL <https://www.statista.com/statistics/1229799/india-smartphone-penetration-rate/>. Last accessed on 26/08/2021.
- [26] Goldman Sachs Global Investment Research. India Internet: A Closer Look Into The Future, 2020. URL https://nextbn.ggvc.com/wp-content/uploads/2020/09/July-2020-GS-India-Internet_-A-Closer-Look-Into-the-Future.pdf. Last accessed on 26/08/2021.
- [27] AT&T Ericsson. Ericsson Mobility Report, November 2020. URL <https://www.ericsson.com/4adc87/assets/local/mobility-report/documents/2020/november-2020-ericsson-mobility-report.pdf>. Last accessed on 26/08/2021.
- [28] CyberMedia Research. Perceived benefits of using smartphones across india as of december 2019, 2019. URL <https://www.statista.com/statistics/1116526/india-perceived-benefits-of-using-smartphones/>. Last accessed on 26/08/2021.
- [29] Statscounter.com. Mobile operating system market share india, 2020. URL <https://gs.statcounter.com/os-market-share/mobile/india/#yearly-2012-2020>. Last accessed on 26/08/2021.
- [30] Ye Zhao, Ngoc Do, Shu-Ting Wang, Cheng-Hsin Hsu, and Nalini Venkatasubramanian. Enabling Offline Access to Facebook Streams on Mobile Devices. In *Proceedings Demo & Poster Track of ACM/IFIP/USENIX International Middleware Conference, MiddlewareDPT '13*, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450325493. doi: 10.1145/2541614.2541622. URL <https://doi.org/10.1145/2541614.2541622>.
- [31] Max Roser and Esteban Ortiz-Ospina. Literacy. *Our World in Data*, 2016. <https://ourworldindata.org/literacy>.
- [32] Sylvia Scribner. Literacy in three metaphors. *American Journal of Education*, 93(1):6–21, 1984. ISSN 01956744, 15496511. URL <http://www.jstor.org/stable/1085087>.

- [33] D. Wagner. Literacy assessment in the third world: An overview and proposed schema for survey use. *Comparative Education Review*, 34:112 – 138, 1990.
- [34] Louis Leung. Effects of Internet Connectedness and Information Literacy on Quality of Life. *Social Indicators Research: An International and Interdisciplinary Journal for Quality-of-Life Measurement*, 98(2):273–290, September 2010. doi: 10.1007/s11205-009-9539-1. URL <https://ideas.repec.org/a/spr/soinre/v98y2010i2p273-290.html>.
- [35] Ministry of Social Justice and Empowerment (India). Literacy rates among scheduled caste and total population in India between 1961 and 2011, 2016. URL <https://www.statista.com/statistics/702170/scheduled-caste-literacy-rate-india/>.
- [36] Literacy rate, adult total (% of people ages 15 and above), 2020. URL <https://data.worldbank.org/indicator/SE.ADT.LITR.ZS>. Last accessed on 26/08/2021.
- [37] Census India, 2011. URL https://censusindia.gov.in/2011census/censusinfodashboard/stock/downloads/Profiles_6/PDF/IND_6.pdf. Last accessed on 26/08/2021.
- [38] Vijesh V. Krishna, Lagesh M. Aravalath, and Surjit Vikraman. Does caste determine farmer access to quality information? *PLOS ONE*, 14(1):1–22, 01 2019. doi: 10.1371/journal.pone.0210721. URL <https://doi.org/10.1371/journal.pone.0210721>.
- [39] Census of India, Language, 2011. URL https://censusindia.gov.in/2011Census/C-16_25062018_NEW.pdf. Last accessed on 28/08/2021.
- [40] Mahendra Dev. Agricultural Reforms in India. *Indian Public Policy Review*, 2 (1 (Jan-Feb)):16–28, 2021.
- [41] National Council of Applied Economic Research. Study on Agricultural Diagnostics for the State of Bihar in India, 2019. URL https://assets.publishing.service.gov.uk/media/5e58f1e0d3bf7f06ffcfec7d/Bihar-Nov_Final_Rev-nov.pdf. Last accessed on 28/08/2021.
- [42] World Bank. Rural population (% of total population), 2018. URL <https://data.worldbank.org/indicator/SP.RUR.TOTL.ZS>. Last accessed on 26/08/2021.
- [43] Sheetal K Agarwal, Arun Kumar, Amit Anil Nanavati, and Nitendra Rajput. Content creation and dissemination by-and-for users in rural areas. In *2009 International Conference on Information and Communication Technologies and Development (ICTD)*, pages 56–65, 2009. doi: 10.1109/ICTD.2009.5426702.

- [44] Sheetal Agarwal, Dipanjan Chakraborty, Swati Challa, Nandakishore Kambhatla, Arun Kumar, Sougata Mukherjea, Amit Anil Nanavati, and Niten-dra Rajput. Pyr.Mea.IT: Permeating IT towards the base of the pyra-mid. *SIGOPS Oper. Syst. Rev.*, 42(1):108–109, January 2008. ISSN 0163-5980. doi: 10.1145/1341312.1341336. URL <https://doi.org/10.1145/1341312.1341336>.
- [45] Arun Kumar, Sheetal K. Agarwal, and Priyanka Manwani. The Spoken Web Application Framework: User Generated Content and Service Cre-ation through Low-End Mobiles. In *Proceedings of the 2010 Interna-tional Cross Disciplinary Conference on Web Accessibility (W4A)*, W4A '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300452. doi: 10.1145/1805986.1805990. URL <https://doi.org/10.1145/1805986.1805990>.
- [46] Sheetal K. Agarwal, Anupam Jain, Arun Kumar, Amit A. Nanavati, and Niten-dra Rajput. The Spoken Web: A Web for the Underprivileged. *SIGWEB Newsl.*, (Summer), June 2010. ISSN 1931-1745. doi: 10.1145/1796390.1796391. URL <https://doi.org/10.1145/1796390.1796391>.
- [47] Sheetal K. Agarwal, Anupam Jain, Arun Kumar, Priyanka Manwani, and Ni-tendra Rajput. Spoken Web: Creation, Navigation and Searching of Voic-eSites. In *Proceedings of the 16th International Conference on Intelligent User Interfaces*, IUI '11, page 431–432, New York, NY, USA, 2011. Asso-ciation for Computing Machinery. ISBN 9781450304191. doi: 10.1145/1943403.1943484. URL <https://doi.org/10.1145/1943403.1943484>.
- [48] Neil Patel, Deepti Chittamuru, Anupam Jain, Paresh Dave, and Tapan S. Parikh. Avaaj Otalo: A Field Study of an Interactive Voice Forum for Small Farmers in Rural India. In *Proceedings of the SIGCHI Conference on Human Fac-tors in Computing Systems*, CHI '10, page 733–742, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605589299. doi: 10.1145/1753326.1753434. URL <https://doi.org/10.1145/1753326.1753434>.
- [49] Indrani Medhi-Thies, Pedro Ferreira, Nakull Gupta, Jacki O'Neill, and Edward Cutrell. KrishiPustak: A Social Networking System for Low-Literate Farmers. CSCW '15, page 1670–1681, New York, NY, USA, 2015. Association for Com-puting Machinery. ISBN 9781450329224. doi: 10.1145/2675133.2675224. URL <https://doi.org/10.1145/2675133.2675224>.
- [50] UNESCO UIS. Literacy, 2021. URL <http://uis.unesco.org/en/topic/literacy#:~:text=Despite%20the%20steady%20rise%20in,most%20of%20whom%20are%20women>. Last accessed on 26/08/2021.
- [51] Dorte Verner. What factors influence world literacy? Is Africa different? Pol-icy Research Working Paper Series 3496, The World Bank, January 2005. URL <https://ideas.repec.org/p/wbk/wbrwps/3496.html>. Last accessed on 26/08/2021.

- [52] Tanushree Chandra. Literacy in india: The gender and age dimension, 2019. URL https://www.orfonline.org/wp-content/uploads/2019/10/ORF_IssueBrief_322_Literacy-Gender-Age.pdf. Last accessed on 26/08/2021.
- [53] Indrani Medhi Thies. User Interface Design for Low-literate and Novice Users: Past, Present and Future. *Foundations and Trends in Human-Computer Interaction*, 8(1):1–72, 2015. ISSN 1551-3955. doi: 10.1561/11000000047. URL <http://dx.doi.org/10.1561/11000000047>.
- [54] Indrani Medhi, Somani Patnaik, Emma Brunskill, S.N. Nagasena Gautama, William Thies, and Kentaro Toyama. Designing Mobile Interfaces for Novice and Low-Literacy Users. *ACM Trans. Comput.-Hum. Interact.*, 18(1), May 2011. ISSN 1073-0516. doi: 10.1145/1959022.1959024. URL <https://doi.org/10.1145/1959022.1959024>.
- [55] Moussa Doumbouya, Lisa Einstein, and Chris Piech. Using Radio Archives for Low-Resource Speech Recognition: Towards an Intelligent Virtual Assistant for Illiterate Users. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14757–14765, 2021.
- [56] Fang Qiao, Jahanzeb Sherwani, and Roni Rosenfeld. Small-Vocabulary Speech Recognition for Resource-Scarce Languages. In *Proceedings of the First ACM Symposium on Computing for Development*, ACM DEV '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450304733. doi: 10.1145/1926180.1926184. URL <https://doi.org/10.1145/1926180.1926184>.
- [57] Sébastien Cuendet, Indrani Medhi, Kalika Bali, and Edward Cutrell. VideoKheti: Making video content accessible to low-literate and novice users. pages 2833–2842, 04 2013. doi: 10.1145/2470654.2481392.
- [58] Indrani Medhi, Meera Lakshmanan, Kentaro Toyama, and Edward Cutrell. Some Evidence for the Impact of Limited Education on Hierarchical User Interface Navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 2813–2822, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450318990. doi: 10.1145/2470654.2481390. URL <https://doi.org/10.1145/2470654.2481390>.
- [59] Aishwarya Ratan, Sunandan Chakraborty, Pushkar Chitnis, Kentaro Toyama, Keng Ooi, Matthew Phiong, and Mike Koenig. Managing Microfinance with Paper, Pen and Digital Slate. *ACM International Conference Proceeding Series*, 12 2010. doi: 10.1145/2369220.2369255.
- [60] Madelaine Plauche, Udhyakumar Nallasamy, Joyojeet Pal, Chuck Wooters, and Divya Ramachandran. Speech Recognition for Illiterate Access to Information and Technology. pages 83 – 92, 06 2006. doi: 10.1109/ICTD.2006.301842.

- [61] GSMA Mobile for Development Impact. Case study: Reuters Market Light, 2014. URL <https://www.gsma.com/mobilefordevelopment/wp-content/uploads/2014/12/M4D-Impact-Case-Study-Reuters-Market-Light.pdf>. Last accessed on 01/09/2021.
- [62] Ministry of Agriculture and Farmers Welfare. Kisan Suvidha, 2016. URL <https://play.google.com/store/apps/details?id=in.cdac.bharatd.agriapp>. Last accessed on 01/09/2021.
- [63] Patidar Technology. Mandi Bhav, 2020. URL <https://play.google.com/store/apps/details?id=com.mandibhav.app&hl=en&gl=US>. Last accessed on 01/09/2021.
- [64] Victor Papanek and R Buckminster Fuller. *Design for the real world*. Thames and Hudson London, 1972.
- [65] Don Norman. *The design of everyday things: Revised and expanded edition*. Basic books, 2013.
- [66] Alan Cooper, Robert Reimann, David Cronin, and Christopher Noessel. *About face: the essentials of interaction design*. John Wiley & Sons, 2014.
- [67] Petr Kosina. Mohen Singh, 2010. URL <https://flic.kr/p/8hc9wF>. Last accessed on 26/08/2021.
- [68] Petr Kosina. Chamkaur Singh, 2010. URL <https://flic.kr/p/8hfv1N>. Last accessed on 26/08/2021.
- [69] Charles. Indian market trader, 2014. URL <https://flic.kr/p/reCZsa>. Last accessed on 26/08/2021.
- [70] Mario Micklisch. Indian truck driver, 2016. URL <https://flic.kr/p/NNGCXd>. Last accessed on 26/08/2021.
- [71] Retsef Levi, Manoj Rajan, Somya Singhvi, and Yanchong Zheng. The impact of unifying agricultural wholesale markets on prices and farmers' profitability. *Proceedings of the National Academy of Sciences*, 117(5):2366–2371, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1906854117. URL <https://www.pnas.org/content/117/5/2366>.
- [72] IDEO Org. The Field Guide to Human-Centered Design. *São Francisco*, 2015. URL <https://www.designkit.org/resources/1>. Last accessed on 26/08/2021.
- [73] Material design, 2021. URL <https://material.io/>. Last accessed on 26/08/2021.
- [74] Dmitriy Rabetckiy. A single-activity android application. why not?!, 2021. URL <https://medium.com/rosberryapps/fa2a5458a099>. Last accessed on 26/08/2021.

-
- [75] Rob Pridham. Modernising a legacy android app architecture, part three: Applying the refactor, 2021. URL <https://robpridham.medium.com/d9d826088427>. Last accessed on 26/08/2021.
- [76] Pierre-Yves Ricau. Advocating against android fragments, 2021. URL <https://medium.com/square-corner-blog/81fd0b462c97>. Last accessed on 26/08/2021.
- [77] Material design - top app bar, 2021. URL <https://material.io/components/app-bars-top>. Last accessed on 26/08/2021.
- [78] Material design - bottom navigation, 2021. URL <https://material.io/components/bottom-navigation>. Last accessed on 26/08/2021.
- [79] Android fragment container view, 2021. URL <https://developer.android.com/reference/androidx/fragment/app/FragmentManager.FragmentContainerView>.
- [80] Firebase realtime database, 2021. URL <https://firebase.google.com/docs/database>. Last accessed on 26/08/2021.
- [81] Firebase open source, 2021. URL <https://firebaseopensource.com/projects/firebase/firebaseui-android/database/readme/>. Last accessed on 26/08/2021.
- [82] Fcainfoweb.nic.in, 2021. URL https://fcainfoweb.nic.in/Reports/Report_Menu_Web.aspx. Last accessed on 26/08/2021.
- [83] Google geocoding api, 2021. URL <https://developers.google.com/maps/documentation/geocoding/overview>. Last accessed on 26/08/2021.
- [84] Greg Bensinger. Google redraws the borders on maps depending on who's looking, 2021. URL <https://www.washingtonpost.com/technology/2020/02/14/google-maps-political-borders/>. Last accessed on 26/08/2021.
- [85] Selenium, 2021. URL <https://www.selenium.dev/>. Last accessed on 26/08/2021.
- [86] Firebase, 2021. URL <https://firebase.google.com/docs/database/web/read-and-write>. Last accessed on 26/08/2021.
- [87] Firebase, 2021. URL <https://firebase.google.com/docs/database/web/structure-data>. Last accessed on 26/08/2021.
- [88] Firebase, 2021. URL <https://firebase.google.com/>. Last accessed on 26/08/2021.
- [89] Joseph Kiesecker. Indian grains, 2020. URL <https://flic.kr/p/2iNkvYS>. Last accessed on 26/08/2021.
-

-
- [90] Imageoptim, 2021. URL <https://imageoptim.com/mac>. Last accessed on 26/08/2021.
- [91] Anychart android, 2021. URL <https://github.com/AnyChart/AnyChart-Android>. Last accessed on 26/08/2021.
- [92] Android drawable resources - layer list, 2021. URL <https://developer.android.com/guide/topics/resources/drawable-resource#LayerList>. Last accessed on 26/08/2021.
- [93] Windows developer standard icons, 2021. URL <https://docs.microsoft.com/en-gb/windows/win32/uxguide/vis-std-icons>. Last accessed on 26/08/2021.
- [94] Google fonts, 2021. URL <https://fonts.google.com/icons?selected=Material+Icons&icon.query=information>. Last accessed on 26/08/2021.
- [95] Apple developer: Human interface guidelines, 2021. URL <https://developer.apple.com/design/human-interface-guidelines/macos/overview/themes/>. Last accessed on 26/08/2021.
- [96] Flags are not languages. Iconography for translations: best practice for communicating availability of translated content, 2014. URL <http://www.flagsarenotlanguages.com/blog/iconography-for-translations-best-practice-for-communicating-availability-of-translated-content/>. Last accessed on 28/08/2021.
- [97] Viewpager2, 2021. URL <https://developer.android.com/jetpack/androidx/releases/viewpager2>. Last accessed on 26/08/2021.
- [98] Dialogflow, 2021. URL <https://cloud.google.com/dialogflow>. Last accessed on 26/08/2021.
- [99] Node.js, 2021. URL <https://nodejs.org/en/>. Last accessed on 26/08/2021.
- [100] Dialogflow inline editor, 2021. URL <https://cloud.google.com/dialogflow/es/docs/fulfillment-inline-editor>. Last accessed on 26/08/2021.
- [101] Google Cloud Platform. Dialogflow Integrations, 2021. URL <https://github.com/GoogleCloudPlatform/dialogflow-integrations>. Last accessed on 26/08/2021.
- [102] Alan Wood's Unicode Resources, 2016. URL <http://www.alanwood.net/unicode/devanagari.html>. Last accessed on 26/08/2021.
- [103] Google Compact Language Detector V3, 2021. URL <https://github.com/google/cld3>. Last accessed on 26/08/2021.
- [104] Mobile Mandi, Google Play Store, 2021. URL <https://play.google.com/store/apps/details?id=com.jaskiratchahal.firebaseio>. Last accessed on 26/08/2021.

- [105] Firebase Dynamic Links, 2021. URL <https://firebase.google.com/docs/dynamic-links>. Last accessed on 26/08/2021.
- [106] Android App Links, 2021. URL <https://developer.android.com/training/app-links>. Last accessed on 26/08/2021.
- [107] Selenium IDE, 2021. URL <https://www.selenium.dev/selenium-ide/>. Last accessed on 26/08/2021.
- [108] Speechrecognizer, 2021. URL <https://developer.android.com/reference/android/speech/SpeechRecognizer>. Last accessed on 26/08/2021.
- [109] Google Cloud Speech To Text, 2021. URL <https://cloud.google.com/speech-to-text/docs/languages>. Last accessed on 26/08/2021.
- [110] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
- [111] Google Cloud Text To Speech, 2021. URL <https://cloud.google.com/text-to-speech/docs/voices>. Last accessed on 26/08/2021.
- [112] Google Cloud Languages, 2021. URL <https://cloud.google.com/dialogflow/es/docs/reference/language>. Last accessed on 26/08/2021.
- [113] Akhtarul Wasey. 50th Report Of The Commissioner For Linguistic Minorities In India, 2014.
- [114] Unicode Language Identifiers and BCP47, 2021. URL <https://util.unicode.org/UnicodeJsps/languageid.jsp>. Last accessed on 26/08/2021.
- [115] Naveena Karusala, Aditya Vishwanath, Aditya Vashistha, Sunita Kumar, and Neha Kumar. “Only If You Use English You Will Get to More Things”: Using Smartphones to Navigate Multilingualism, page 1–14. Association for Computing Machinery, New York, NY, USA, 2018. ISBN 9781450356206. URL <https://doi.org/10.1145/3173574.3174147>.
- [116] Kentaro Toyama. Designing for Aspirations. *Interactions*, 27(3):61–63, April 2020. ISSN 1072-5520. doi: 10.1145/3394059. URL <https://doi.org/10.1145/3394059>.
- [117] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [118] Christopher Olah. Understanding LSTM Networks, 2015. URL <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Last accessed on 30/08/2021.

- [119] Apple Frameworks Natural Language Processing Team. Can Global Semantic Context Improve Neural Language Models?, September 2018. URL <https://machinelearning.apple.com/research/can-global-semantic-context-improve-neural-language-models>. Last accessed on 30/08/2021.
- [120] Facebook Engineering Alexander Sidorov. Transitioning entirely to neural machine translation, August 2017. URL <https://engineering.fb.com/2017/08/03/ml-applications/transitioning-entirely-to-neural-machine-translation/>. Last accessed on 30/08/2021.
- [121] Abigail See. CS224N Lecture 7: Vanishing Gradients and Fancy RNNs, 2019. URL <https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture07-fancy-rnn.pdf>. Last accessed on 30/08/2021.
- [122] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. A comparison of ARIMA and LSTM in forecasting time series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1394–1401. IEEE, 2018.
- [123] Sudeshna Sarkar. Cs60010: Deep learning, recurrent neural network, February 2018. URL <https://cse.iitkgp.ac.in/~sudeshna/courses/DL18/lec15-LSTM-12-Feb-18.pdf>. Last accessed on 30/08/2021.
- [124] Recurrent Neural Networks (RNN) with Keras, 2021. URL <https://www.tensorflow.org/guide/keras/rnn>. Last accessed on 30/08/2021.
- [125] CSC Sekhar, Devesh Roy, and Yogesh Bhatt. Food inflation and volatility in india: trends and determinants. *Indian Economic Review*, 53(1):65–91, 2018.
- [126] Carol M Barnum. *Usability testing essentials: ready, set... test!* Morgan Kaufmann, 2020.
- [127] Jakob Nielsen. Ten usability heuristics, 1994. Last accessed on 26/08/2021.
- [128] Whitney Quesenbery. Balancing the 5Es of usability. *Cutter IT Journal*, 17(2): 4–11, 2004.
- [129] Leo A Goodman. Snowball sampling. *The annals of mathematical statistics*, pages 148–170, 1961.
- [130] Amy Ellard-Gray, Nicole K. Jeffrey, Melisa Choubak, and Sara E. Crann. Finding the Hidden Participant: Solutions for Recruiting Hidden, Hard-to-Reach, and Vulnerable Populations. *International Journal of Qualitative Methods*, 14(5):1609406915621420, 2015. doi: 10.1177/1609406915621420. URL <https://doi.org/10.1177/1609406915621420>.
- [131] Nicola Dell, Vidya Vaidyanathan, Indrani Medhi, Edward Cutrell, and William Thies. “Yours is Better!”: Participant Response Bias in HCI, page 1321–1330. Association for Computing Machinery, New York, NY, USA, 2012. ISBN 9781450310154. URL <https://doi.org/10.1145/2207676.2208589>.

-
- [132] UXCam App Analytics, 2021. URL <https://uxcam.com/>. Last accessed on 24/08/2021.
- [133] John Brooke et al. SUS - A quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.
- [134] James R. Lewis. IBM computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. *International Journal of Human–Computer Interaction*, 7(1):57–78, 1995. doi: 10.1080/10447319509526110. URL <https://doi.org/10.1080/10447319509526110>.
- [135] Bettina Laugwitz, Theo Held, and Martin Schrepp. Construction and Evaluation of a User Experience Questionnaire. In Andreas Holzinger, editor, *HCI and Usability for Education and Work*, pages 63–76, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-89350-9. doi: 10.1007/978-3-540-89350-9_6. URL https://doi.org/10.1007/978-3-540-89350-9_6.
- [136] Aaron Bangor, Philip T. Kortum, and James T. Miller. An Empirical Evaluation of the System Usability Scale. *International Journal of Human–Computer Interaction*, 24(6):574–594, 2008. doi: 10.1080/10447310802205776. URL <https://doi.org/10.1080/10447310802205776>. Last accessed on 26/08/2021.
- [137] Jeff Sauro and James R Lewis. *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016. URL <https://doi.org/10.1016/C2010-0-65192-3>. Last accessed on 26/08/2021.
- [138] James R. Lewis. Psychometric Evaluation of the PSSUQ Using Data from Five Years of Usability Studies. *International Journal of Human–Computer Interaction*, 14(3-4):463–488, 2002. doi: 10.1080/10447318.2002.9669130. URL <https://doi.org/10.1080/10447318.2002.9669130>. Last accessed on 26/08/2021.
- [139] User Experience Questionnaire, 2021. URL <https://www.ueq-online.org/>. Last accessed on 23/08/2021.
- [140] Joey Benedek and Trish Miner. Measuring Desirability: New methods for evaluating desirability in a usability lab setting. *Proceedings of Usability Professionals Association*, 2003(8-12):57, 2002.
- [141] Firebase Test Lab, 2021. URL <https://firebase.google.com/docs/test-lab/android/robo-ux-test>. Last accessed on 23/08/2021.
- [142] UI/Application Exerciser Monkey, 2021. URL <https://developer.android.com/studio/test/monkey>. Last accessed on 23/08/2021.
- [143] Priyam Patel, Gokul Srinivasan, Sydur Rahaman, and Iulian Neamtii. On the effectiveness of random testing for android: or how i learned to
-

- stop worrying and love the monkey. In *Proceedings of the 13th International Workshop on Automation of Software Test*, pages 34–37, 2018. doi: 10.1145/3194733.3194742. URL <https://dl.acm.org/doi/10.1145/3194733.3194742>.
- [144] Android Profiler, 2021. URL <https://developer.android.com/studio/profile/android-profiler>. Last accessed on 23/08/2021.
- [145] Location Request - Google Play Services, 2021. URL <https://developers.google.com/android/reference/com/google/android/gms/location/LocationRequest.html>. Last accessed on 23/08/2021.
- [146] Punjab Crop Reporting Service. Punjab Crop List, 2021. URL http://crs.agripunjab.gov.pk/crop_details. Last accessed on 31/08/2021.
- [147] Information Commissioner’s Office. Guide to Privacy and Electronic Communications Regulations, 2021. URL <https://ico.org.uk/for-organisations/guide-to-pecr/communications-networks-and-services/location-data/>. Last accessed on 31/08/2021.
- [148] Android. App permissions best practices, 2021. URL <https://developer.android.com/training/permissions/usage-notes>. Last accessed on 31/08/2021.
- [149] Google Cloud. Encryption at rest in Google Cloud, 2021. URL <https://cloud.google.com/security/encryption/default-encryption>. Last accessed on 31/08/2021.
- [150] Google Cloud. Encryption in Transit in Google Cloud, 2021. URL <https://cloud.google.com/security/encryption-in-transit>. Last accessed on 31/08/2021.
- [151] UXCam. Our approach to privacy and security, 2021. URL <https://help.uxcam.com/hc/en-us/articles/360017823571>. Last accessed on 31/08/2021.
- [152] Waze Navigation, 2021. URL <https://www.waze.com/>. Last accessed on 24/08/2021.

Appendix A

System Usability Scale

	Strongly disagree						Strongly agree
1. I think that I would like to use this system frequently	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
2. I found the system unnecessarily complex	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
3. I thought the system was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
4. I think that I would need the support of a technical person to be able to use this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
5. I found the various functions in this system were well integrated	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
6. I thought there was too much inconsistency in this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
7. I would imagine that most people would learn to use this system very quickly	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
8. I found the system very cumbersome to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
9. I felt very confident using the system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		
10. I needed to learn a lot of things before I could get going with this system	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		
	1	2	3	4	5		

Appendix B

Post-Study System Usability Questionnaire

1. Overall, I am satisfied with how easy it is to use this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

2. It was simple to use this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

3. I could effectively complete the tasks and scenarios using this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

4. I was able to complete the tasks and scenarios quickly using this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

5. I was able to efficiently complete the tasks and scenarios using this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

6. I felt comfortable using this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

7. It was easy to learn to use this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

8. I believe I could become productive quickly using this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

9. The system gave error messages that clearly told me how to fix problems.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

10. Whenever I made a mistake using the system, I could recover easily and quickly.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

11. The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

12. It was easy to find the information I needed.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

13. The information provided for the system was easy to understand.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

14. The information was effective in helping me complete the tasks and scenarios.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

15. The organization of information on the system screens was clear.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

Note: *The interface includes those items that you use to interact with the system. For example, some components of the interface are the keyboard, the mouse, the screens (including their use of graphics and language).*

16. The interface of this system was pleasant.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

17. I liked using the interface of this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

18. This system has all the functions and capabilities I expect it to have.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

19. Overall, I am satisfied with this system.

STRONGLY									STRONGLY
AGREE	1	2	3	4	5	6	7		DISAGREE

COMMENTS:

Appendix C

Usability Evaluation Questionnaire

	1	2	3	4	5	6	7		
annoying	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	enjoyable	1
not understandable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	understandable	2
creative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	dull	3
easy to learn	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	difficult to learn	4
valuable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	inferior	5
boring	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	exciting	6
not interesting	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	interesting	7
unpredictable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	predictable	8
fast	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	slow	9
inventive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	conventional	10
obstructive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	supportive	11
good	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	bad	12
complicated	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	easy	13
unlikable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	pleasing	14
usual	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	leading edge	15
unpleasant	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	pleasant	16
secure	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	not secure	17
motivating	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	demotivating	18
meets expectations	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	does not meet expectations	19
inefficient	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	efficient	20
clear	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	confusing	21
impractical	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	practical	22
organized	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	cluttered	23
attractive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unattractive	24
friendly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	unfriendly	25
conservative	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	innovative	26

Appendix D

Product Reaction Cards

Simplistic	Inviting	Clean	Irrelevant	Patronizing
Not Valuable	Approachable	Dated	Valuable	Consistent
Boring	Effortless	Comprehensive	Stable	Easy to use
Motivating	Compelling	Overbearing	Disconnected	Satisfying
Organized	Fragile	Accessible	Confusing	Useful
Fresh	Creative	Relevant	Impressive	Ordinary
Energetic	Not Secure	Low Maintenance	Stimulating	Enthusiastic
Empowering	Unconventional	Controllable	Exceptional	Predictable
Desirable	Comfortable	Impersonal	Business-like	Convenient
Effective	Difficult	Frustrating	Clear	Gets in the way
Powerful	Customizable	Hard to Use	Fast	Stressful
Time-Saving	Connected	Compatible	Calm	Undesirable
Attractive	Efficient	Poor quality	Inconsistent	Uncontrollable
Familiar	Overwhelming	Unpredictable	Complex	Confident
Unrefined	Rigid	Engaging	Annoying	Busy
Expected	Sterile	Advanced	Essential	Straight Forward
Unapproachable	Distracting	Meaningful	Trustworthy	Old
Intuitive	Cutting edge	Integrated	Unattractive	Intimidating
Time-consuming	Secure	Ineffective	Helpful	Too Technical
Optimistic	Personal	Exciting	Professional	High quality
Disruptive	Collaborative	Fun	Entertaining	Flexible
Inspiring	Slow	Appealing	Understandable	Incomprehensible
Dull	Responsive	Reliable	Sophisticated	
Innovative	Novel	Usable	Friendly	

Developed by and © 2002 Microsoft Corporation. All rights reserved.